ABSTRACT

Title of dissertation: EFFICIENT SENSING, SUMMARIZATION AND CLASSIFICATION OF VIDEOS

Nitesh Shroff, Doctor of Philosophy, 2012

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

Motion perception is an integral part of visual information processing. For example, humans use motion to perceive shape and structure of a scene, segment and recognize objects. Similarly, in computational vision, motion cues have been extensively used in numerous applications e.g., reconstructing 3D structure, object segmentation, etc. But there are several other applications such as pose estimation, scene recognition, etc., where motion plays a unique role, but traditionally they have been studied using cues other than motion. In this dissertation, we study few such applications with a focus on characterizing the role of motion. In particular, we study the role of motion in efficient (a) sensing, (b) summarization, and (c) classification of videos.

We start by developing efficient sensing techniques, particularly in cases where computational vision is used for measurement – inferring depth, position, orientation, etc. of the scene elements. Towards this direction, we begin with the goal of devising sensing techniques that allows the estimation of the scene layout of a

generic scene i.e., the depth map of a scene. This is achieved by proposing an architecture and algorithm that senses the video by varying focal settings between consecutive frames. By extending the paradigm of Depth-from-defocus (DFD) to dynamic scenes, we achieve the reconstruction of the depth video and all-focus video from the captured video. This is followed by devising a technique which under constrained scenarios allows us to take a step further and estimate the precise location and orientation of the objects in the scene. We show that by capturing a sequence of images, while moving the illumination source between two consecutive frames, we can extract specular features on the high-curvature metallic objects. Robustly extracted specular features then allow us to estimate the pose of the objects with applications in machine vision.

Next, we address the problem of concisely representing large video data. The goal here is to gain a quick overview of the video with minimum loss of details. We argue that this can be achieved by optimizing for the following two conflicting criteria: (a) Coverage – requires that the summary be able to represent the original video well, and (b) Diversity – requires that the elements of the summary be as distinct from each other as possible. This is formulated as a subset selection problem first in the Euclidean space and then generalized to non-Euclidean manifolds. The generic non-Euclidean manifold formulation allows the algorithm to handle generic computer-vision datasets like shapes, textures, linear dynamical systems, etc. A novel annealing-based alternation algorithm is proposed to select the optimal subset. Our experimental evaluation convincingly demonstrates that this formulation, effectively highlights diverse motion patterns in the video and hence outputs good

summaries without actually using any domain knowledge.

Finally, we turn our attention to classification of videos. Here, we begin with devising exact and approximate nearest neighbor (NN) techniques for fast retrieval of videos from large databases. As these videos or their representations, lie in non-Euclidean manifolds, the focus here is on formulating the problem such that it utilizes the geometry of the space. We present a geodesic hashing technique which employs intrinsic geodesic based functions to hash the data for realizing approximate but fast nearest neighbor retrieval. The proposed family of hashing functions, although intrinsic, is optimally selected to empirically satisfy the Locality Sensitive Hashing property. This work is followed up by another classification technique which focuses on generating content-based, particularly scene-based, annotations of videos. We focus on characterizing the motion of scene elements, and show that it not only provides fine-grained description of videos but also improves the classification accuracy. Subsequently, we propose dynamic attributes which can be augmented with spatial attributes of a scene to categorize dynamic scenes in a semantically meaningful way.

EFFICIENT SENSING, SUMMARIZATION AND CLASSIFICATION OF VIDEOS

by

Nitesh Shroff

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2012

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Larry Davis
Professor David Jacobs
Professor Min Wu
Professor Pavan Turaga

To my *parents*

Acknowledgments

First and foremost, I want to thank God!

I owe my deepest gratitude to my advisor Prof. Rama Chellappa for his guidance and mentoring over the years. I have learned immensely from Prof. Chellappa about research and life. His dedication to work, outlook towards research, and extremely high standards that he sets for himself has been thoroughly motivating for me at various stages of my doctoral studies. His mentoring has been paramount in this dissertation taking its shape and his guidance has enormously impacted my developing a better and broader perspective towards research. He has also taught and shown by example that if we love what we do, we can scale any professional heights.

It is an honor for me to have Prof. Larry Davis, Prof. Min Wu, Prof. David Jacobs and Prof. Pavan Turaga on my dissertation committee. I am very thankful to them for serving on my committee and providing valuable feedback on this dissertation. I am also very thankful to Prof. André Tits who introduced me to the mathematical rigor through his course on Optimal Control and then later gave me an opportunity to co-teach Signals and Systems with him.

I am deeply indebted to Prof. Ashok Veeraraghavan, Prof. Pavan Turaga and Dr. Dikpal Reddy whose mentoring since my early days as a graduate student has been tremendously valuable to me. From them, I have learned various aspects of conducting research – developing intuition, picking a problem, proposing solutions and designing appropriate experiments.

This thesis would not have been possible without the great mentors that I have had at various times during the course of my PhD – Dr. Yuichi Taguchi, Dr. Oncel Tuzel, Dr. Mainak Sen, Dr. Srikumar Ramalingam, Dr. Mahesh Ramachandran, and Dr. Aswin Sankaranarayanan.

I would like to show my gratitude to my fellow group members – Dr. Ming-Yu Liu, Dr. Raghuraman Gopalan, Sima Taheri, Dr. Kaushik Mitra, Ashish Shrivastava, Dr. Anne Jorstad, Sumit Shekhar, Jaishanker Pillai, Dr. Vishal Patel, Dr. Ruonan Li, Hien Nguyen, Garrett Warnell, and Ravi Teja for providing me feedback at various stages and teaching me numerous things about computer vision.

I am eternally obliged to my friends Preeti Bharagava, Kapil Anand, Shalabh Jain, Kaustubh Jain, Dikpal Reddy, Srinivasa Rangan and Sachin Kadloor. They have made their true support available in numerous ways and have always provided the right motivation and support which has made my graduate studies a very comfortable journey. Thanks are also due to my friends and housemates, who have never let me feel away from home – Avinash Varna, Raghuraman Gopalan, Himanshu Tyagi, Jishnu Keshavan, Balaji Srinivasan, Aparna Kotha, Harita Tenneti, Ashish Misra, Rakesh Garg, Osman Yagan, Ravi Garg, Neeraj Jain, Tarun Singh, and Arun Karthikeyan. I also want to thank my friends from Mandakini hostel, and "Grad Babus" group from IIT Madras. I have been very fortunate to have brothers like Shyam, Gyanesh, and Rohit who have continually provided their support to me. Words are not enough to thank Sophie for always being there with me.

I would also like to thank UMIACS and ECE personnel without whom everything would come to stand still. Special thanks are due to Janice Perrone, Melanie

Prange and Arlene Schenk.

This dissertation would not have been possible without constant support of my parents. Words would not be enough to express my gratitude to them. This dissertation is a testament to their commitment towards my education and sacrifices they have made in shaping up my life. I dedicate this dissertation to them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Videos (or moving images) are a rich source of information with wide applications ranging from entertainment, scientific research to security and surveillance. They render themselves more attractive over still images as they additionally capture scene and object motion information. Motion, which is an important aspect of visual perception, can be simply defined as relative positional changes in time between objects. Several psychophysical studies have shown that human perception largely utilizes motion information to infer about the visible world [5]. Roger and Graham [6] showed that motion parallax acts as a cue to perceive relative depth and shape information of a three dimensional scene. Image regions with different motions correspond to distinct objects and hence provide cues for object segmentation. It has been demonstrated that three-dimensional form of a rigidly moving object can be uniquely specified within an undetermined scale factor [7, 8, 9, 5]. In short, when the observer is stationary, object's motion gives information about its shape and orientation.

On the other hand, a moving observer perceives relation of size and distance of stationary objects. Determining three-dimensional structure from motion has been extensively studied both in psychophysics [10, 11] and computational vision [12, 13]. Furthermore, motion information is known to be useful both at the object-level and

the global-level. At the object-level, many experimental studies have shown that motion information helps by enhancing the recovery of information about shape [14], edges [15], views [16], etc. At a global level, motion information can be used for video compression (e.g. MPEG-2), video indexing [17], etc. In short, it would be impossible to overstate the importance of motion information from the perspectives of both psychophysics and computer vision.

In addition to these well-explored cues that motion provides, there are several other applications such as pose estimation, scene recognition, etc., where motion plays an integral role, but traditionally they have been studied using static appearance cues from single view or sometimes from multiple view. This dissertation focuses on addressing such problems with a focus on characterizing the role of motion in these problems. In particular, this dissertation addresses the role of motion in different aspects of video consumption: (a) sensing, (b) summarization, and (c) classification.

1.1 Sensing

Humans extensively use vision for the perception of scene elements and understanding of the scene layout. The latter capability uses vision-based measurement of several properties of the scene including depth, orientation and location of individual elements. Motivated by this human capability, significant efforts have been made for developing computational vision techniques for measurement of scene elements. Despite great advances, this remains a fundamental problem in computer vision.

The challenges involved here can be largely attributed to the conventional imaging system which maps the three-dimensional world onto to a two-dimensional image or video. This mapping causes loss of the depth information of scene elements, along with the introduction of artifacts like blur and sensor noise. Loss of depth information implies that inferring scene layout from the given image becomes ill-posed and highly ambiguous. Further, introduction of artifacts like blur and noise substantially diminishes the performance of any computer vision algorithm. These limitations, as shown by researchers recently, can largely be addressed by modifications to the traditional sensing process.

In the first part of the dissertation, we pursue this line of thought, wherein the goal is to devise techniques for efficient sensing. Specifically, we start by addressing the problem of sensing a dynamic scene with output as a blur-free video along with the depth information of the scene. It is shown here that if during sensing of a video, the focal setting is changed between two consecutive frames, it enables us to extract depth information along with the all-focus video. Specifically, we explore the paradigm of Depth-from-defocus (DFD) which exploits blur as a cue to extract depth information. The setting of DFD has traditionally been constrained to the static scene which prohibits its application to dynamic scenes. We address this limitation and extend the paradigm of DFD to dynamic scenes. Here, we show that motion information, indeed, helps us in establishing correspondence, thus allowing the estimation of depth map.

Depth information, although immensely useful, is restrictive when it comes to using computational vision for precise interaction with the scene. For instance,

consider the case where a robot needs to pick up an object. In this case, the robot needs to accurately estimate the location and orientation of the object before it can interact with it. Such precise measurements are common in machine vision applications where image sensors are used for industrial automation. In the next chapter of the dissertation, we address one such machine vision problem where the goal is to accurately estimate the pose of metallic objects placed in a bin. Here again, it is shown that motion provides an important cue regarding the curvature of the object.

Perceived motion in a sequence of images is mainly caused due to the motion of scene elements. But it can also be caused by the motion of non-scene elements like the camera, illumination sources, etc. Characterizing motion due to the non-scene elements has been shown to provide crucial information about the scene. For instance, in the well-studied problem of structure-from-motion, camera is moved to sense the static scene to enable its 3D reconstruction. Similarly, structured light utilizes a sequence of projected structure [18] on the scene to determine the depth of the scene elements. In our work, we focus on the effect caused by the motion of illumination source. We show that sensing of a sequence of images by moving the illumination source between subsequent captures leads to robust extraction of novel specular features on high-curvature metallic surfaces. Features, once extracted, are then used to estimate the pose of such objects.

1.2 Summarization

Recent years have seen an explosive growth in the number of videos being generated and shared on the Internet. Handling such large amount of video data along with its efficient consumption has necessitated the development of techniques for concise representation of videos. In the second part of this dissertation, we address this problem of concisely representing videos.

We start by seeking answers to the question 'What comprises a good summary?'. One possible answer is using rules / knowledge of the domain to create a good summary. But, this would require identifying domains of each video and independently developing rules for each domain. Rather, a more attractive solution is to seek domain-agnostic solution which can be applied to any video independent of its domain. Towards this direction, we develop a mathematical framework to concisely represent videos with an objective to gain a quick overview of the video while minimizing the loss of details. The problem is formulated as a subset selection problem with the objective to select representative yet diverse exemplars. Our experimental evaluation convincingly demonstrates that this formulation, effectively highlights diverse motion patterns in the video and hence outputs good summaries without actually using any domain knowledge.

One of the key assumption behind this formulation is that the objects or their representation lie in the Euclidean space. This implies that common features/models in computer vision like shapes, bag-of-words, linear dynamical systems, etc. can not be handled by this framework. In the ensuing chapter, we address this limitation and

generalize this framework to be able to handle generic models/features which lie in non-Euclidean space. This requires us to formulate the problem while utilizing the geometry of the underlying space on which features lie. The optimization problem thus formulated is solved by a novel annealing-based alternation algorithm.

1.3 Classification

In the final part of the dissertation, we turn our attention to the classification of unconstrained videos and their retrieval from large databases. We start with devising both exact and approximate nearest neighbor (NN) retrieval techniques. As these videos and/or their representations, lie in non-Euclidean manifolds, the focus here is on formulating the problem such that it utilizes the geometry of the space. Unfortunately, the well-studied problem of nearest-neighbor retrieval has largely focused on datasets lying in Euclidean spaces. This limits the direct application of these techniques for retrieval of videos as the distance between two motion patterns is not accurately characterized by the usual ℓ_2 norm. The other large section in the fast nearest neighbor retrieval literature has been about handling data which lies in a metric space. These techniques, can largely be applied directly for videos. But, this would mean that only the distance between two data-points is utilized while ignoring the underlying geometry of the space.

For the exact search, an efficient tree structure is introduced using diverse clusters. But, if one would be willing to accept a small amount of reduction in accuracy for large gains in speed, efficient approximate NN techniques can also be

developed on non-Euclidean manifolds. Towards this, we devise a geodesic hashing technique which employs intrinsic geodesic based functions to hash dataset. The proposed family of hashing functions, although intrinsic, is optimally selected to empirically satisfy the Locality Sensitive Hashing property.

In an alternate classification technique, we focus on generating content-based annotations for videos. At an abstract level, one would approach this by characterizing motion pattern along with appearance cues. This line of thought has been extensively pursued in the action/event recognition literature where the motion of a human or a group of humans along with their pose information is analyzed. On the other hand, scene recognition, a fundamental problem in computer vision, has been largely restricted to using appearance cues. In the last chapter of this dissertation, we address the problem of characterizing motion information of scene elements in order to obtain fine-grained description of dynamic scenes. Here, we explore accurate and generalizable computational models for characterizing the dynamics of unconstrained dynamic scene videos.

This annotation can also be obtained at a global level by describing dynamic scenes using motion attributes. These motion attributes when augmented with spatial attributes lead to semantically meaningful organization of videos.

Before proceeding further, we briefly introduce the chapters of the dissertation below:

1.4 Variable Focus Video: Reconstructing Depth and Video for Dynamic Scenes

Majority of the cameras around us have auto-focus capability which is highly under-utilized. In this chapter, we show that if utilized appropriately, this has the ability to convert a 30fps conventional camera into a co-located 30fps depth and all-focus video camera. This is achieved by exploring the paradigm of Depth-from-defocus (DFD) which utilizes defocus blur as depth cue. Traditionally, DFD has been constrained to static scenes. In this chapter, we first examine the effect of scene and/or camera motion on traditional DFD. It is shown that motion of scene elements render DFD algorithms to fail as correspondence is no longer satisfied.

In order to address this limitation, we show that if accurate estimate of motion is given, one can robustly warp the focal stack to create a virtual focal stack where DFD algorithms can be applied to obtain depth map. The focus is then turned towards accurate estimation of motion in the presence of varying blur. We discuss how varying blur setting causes inherent biases in the estimation of optical flow explained by violation of brightness constancy assumption. We then propose a robust way to handle this by iterating between optical flow estimation and depth estimation. A prototype is built and experimental results on challenging scenes are demonstrated.

1.5 Bin Picking of Specular Objects

In this chapter, the machine vision problem of bin picking of metallic objects is studied. Despite large advances in machine vision algorithms, tackling objects that do not have a near-Lambertian reflectance is challenging. In this chapter, we show that the effect of specularities need not be treated as a source of noise but rather as a feature that can and does enhance our ability to detect, recognize and interact with specular objects. Specifically, the effect of motion of illumination direction is characterized and is used to propose a novel specular feature. It is shown that when the illumination source moves, the specular highlights remain in a region with radius inversely proportional to the surface curvature. This allows us to robustly and reliably extract features on high-curvature metallic objects using an inexpensive multi-flash camera (MFC).

One can then use multiple views of the object using the MFC in order to triangulate and obtain the 3D location and pose of the shiny objects. Finally, a system is developed consisting of a robot arm with an MFC that can perform automated detection and pose estimation of shiny screws within a cluttered bin, achieving position and orientation errors less than 0.5 mm and 0.8 degrees respectively.

1.6 Video Précis: Highlighting Diverse Aspects of a video

Recent years have seen an explosive growth in the amount of visual data being generated. This has necessitated the ability to gain a quick overview of the contents of a video. In this chapter, we focus on this problem of video abstraction

which requires one to identify key aspects of a video that describes its essence. We propose two criteria, that a summary should have: (a) Coverage, and (b) Diversity. The first criterion suggests that the summary should 'cover' most of the video in terms of global representation. The second criterion suggests that the elements of the summary should be as distinct as possible. A novel formulation is developed to quantify this trade-off using a unified cost function. An algorithm, similar to k-medoid algorithm, is then proposed to optimize the cost function and results are demonstrated on four different classes of videos. We evaluate results both quantitatively and qualitatively. The formulation ensures that the optimal summary captures both the dominant motion and the diverse motion in it.

1.7 Manifold Précis: An Annealing Technique for Diverse Sampling of Manifolds

The framework, presented in the previous chapter, is limited in the following two ways: (a) while most of the features and models in computer vision lie in non-Euclidean space, the framework presented earlier is constrained to handling datasets whose representations lie in the Euclidean space, and (b) the proposed optimization framework is computationally intensive and suffers from the problem of local minima. In this chapter, we address these two limitations by generalizing the Précis framework to non-Euclidean spaces and proposing an efficient optimization technique.

A general theory is formulated which encompasses not just traditional tech-

niques devised for vector spaces, but also non-Euclidean manifolds, thereby enabling these techniques to shapes, human activities, textures and many other image and video based datasets. Intrinsic manifold measures are proposed for measuring the quality of a selection of points with respect to their representative power, and their diversity. This is followed by developing efficient algorithm to optimize the cost function using a novel annealing-based iterative alternation algorithm. The proposed formulation is applicable to manifolds of known geometry as well as to manifolds whose geometry needs to be estimated from samples. Experimental results show the strength and generality of the proposed approach.

1.8 Fast Nearest Neighbor on non-Euclidean Manifolds

Large databases of videos are increasingly becoming common-place due to the growth of personal collections and Internet archives. To make these datasets more easily accessible to users, it is important to develop fast retrieval methods that allow fast retrieval and access to information. In this chapter, we study the problem of fast nearest-neighbor for data lying on non-Euclidean manifolds. A formal treatment of this problem is provided for both exact and approximate nearest neighbor search. First, we develop an efficient tree based structure for exact but fast computation of nearest neighbor. But, exact nearest neighbor, although highly accurate and faster than exhaustive search, is still computationally intensive. However, in several applications, we only require approximate nearest neighbors. Towards this direction, we develop an intrinsic geodesic hashing technique which employs geodesics to hash

the data. The proposed family of hashing functions, although intrinsic, is optimally selected to empirically satisfy the Locality Sensitive Hashing property.

1.9 Moving Vistas: Exploiting Motion for Describing Scenes

Recognizing scenes at a global level is a fundamental problem in computer vision. To a large extent, scene recognition literature has focused on using static appearance cues. But, a lot of times, scenes have characteristic motion information. In this chapter, we study the role of scene dynamics in the improved representation of scenes. Accurate and generalizable computational models for characterizing the dynamics of scenes is explored. Due to the unconstrained nature and lack of in-depth understanding of the physics of these motion, most state-of-the-art models perform poorly. However, chaos-theory, which models dynamics purely based on observation and avoids making any assumption, leads to the best classification accuracy.

As a next step, it is shown that motion information can also be described using motion attributes. These motion attributes characterize the quality of motion in terms of their global properties and provide semantically meaningful organization of videos. These are derived from coarse spatio-temporal properties such as degree of flow-regularity, degree of flow granularity etc. These attributes when combined with spatial attributes provide content-based annotations of the video.

1.10 Contributions

In this dissertation, we make the following contributions:

- We extend Depth-from-defocus (DFD) and Extended Depth-of-Field (EDOF) imaging to dynamic scenes by explicitly accounting for scene and camera motion. We carefully analyze the effects of (a) varying focal blur and (b) occlusions on the motion compensation problem and design an iterative refinement algorithm that explicitly tackles such errors via reblurring. A prototype 30 fps variable depth-of-field and range video camera is implemented and experimental results are shown on several challenging scenes.

- We show that regions of high curvature in highly specular objects produce specular reflections largely independent of the location of the illumination sources, which can be used as a distinct features for object detection and pose estimation. An inexpensive multi-flash camera can be used to reliably detect such specular features. We use screws as an example part and develop two and three-view based algorithms for 3D pose estimation using triangulation. The algorithm is implemented on an industrial robot and very high location and angular accuracy is shown for 3D pose estimation.

- We propose two criteria 'coverage' and 'diversity' that a video summary should embody. A technique for optimizing these two criteria is proposed. The effectiveness of the proposed framework is shown on four different classes of videos and thoroughly evaluated using both quantitative and qualitative measures.

- We present the first formal treatment of subset selection for the general case of manifolds. A novel annealing-based alternation algorithm is proposed to efficiently solve the optimization problem. An extension of the algorithm for

data manifolds is then presented.

- We present an analysis of indexing non-Euclidean spaces whose geometric properties are known. We first deploy an efficient tree structure utilizing the underlying geometry of data. We devise a geodesic hashing technique which employs intrinsic geodesic based functions to hash the data. The proposed family of hashing functions, although intrinsic, is optimally selected to empirically satisfy the Locality Sensitive Hashing property.

- We study the role of dynamics of scene elements for improved representation of scenes, where, it is shown that characteristic motion of scene elements not only provide fine-grained description of the scene but also helps in better recognition accuracy of scenes. The problem of finding accurate and generalizable ways is then addressed for characterizing the hard-to-model dynamics of unconstrained scenes via chaotic systems. We also propose dynamic attributes that describe dynamic scenes at a global level and show semantic organization of videos using these attributes.

Chapter 2

Variable Focus Video: Reconstructing Depth and Video for Dynamic Scenes

2.1 Introduction

Cameras have become ubiquitous with billions of them present in various forms including cell-phone cameras, surveillance cameras, cameras installed on cars, at home, etc. In most cases, these cameras are passive devices just recording and saving video streams. This leaves most of the acquired data unprocessed. A major bottleneck towards automating the visual scene interpretation is the lack of 3D information that is crucial for scene understanding. The goal of this chapter is to make traditional video cameras be able to extract meaningful 3D information by varying the focal distance during the acquisition of the video.

Variable focus makes cameras 3D: Most cameras are already equipped with features such as auto-focus, variable focal length and zoom, all of which require the focal distance to change. Unfortunately, this ability of the camera is significantly under-utilized. The ability to auto-focus is only used to obtain an image in which the subject of interest is in-focus. The fact that different subjects in the scene were in-focus at different focal distances enables the extraction of 3D information from these devices using depth from focus/defocus methods. We exploit this variable

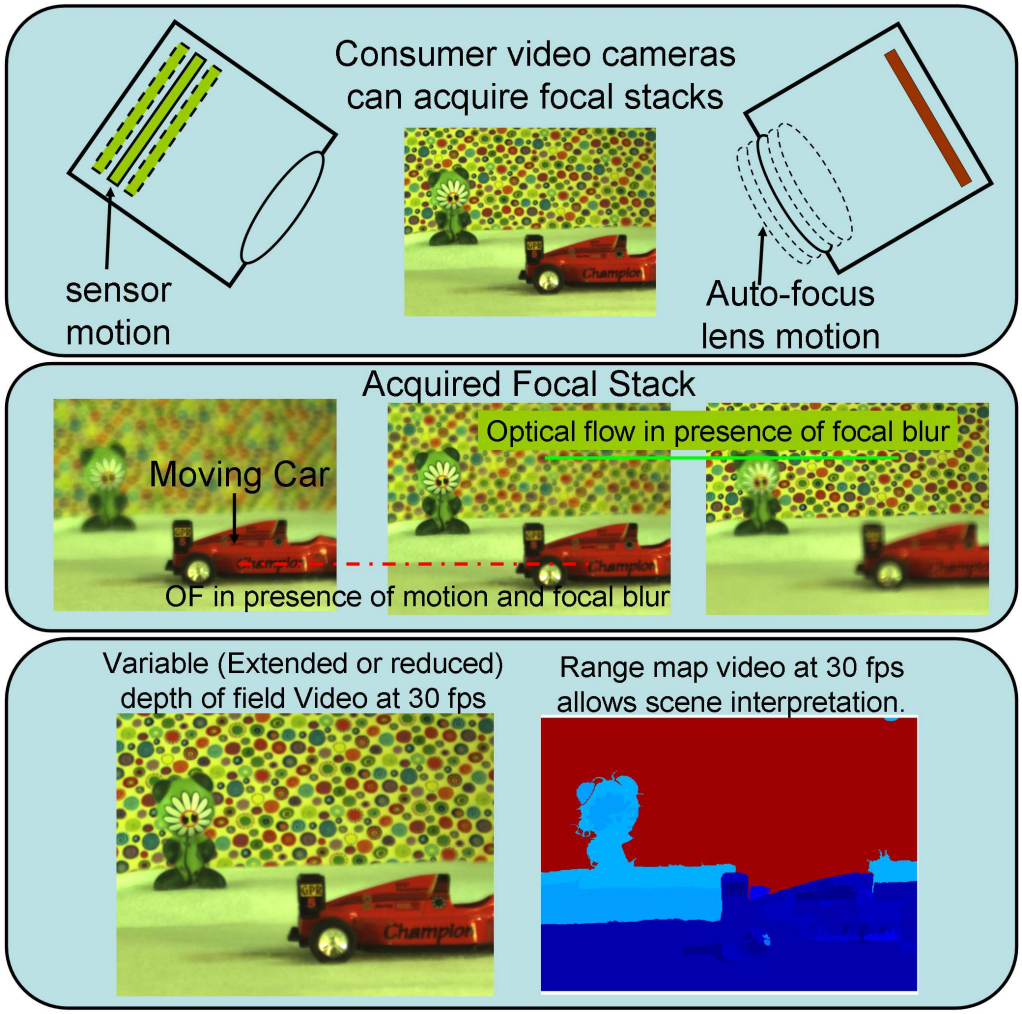


Figure 2.1: Consumer video cameras have the functionality to automatically focus. We can exploit this feature to acquire a focal stack and perform depth from defocus. In this chapter, we show how this can be achieved in the presence of scene and/or camera motion, thereby turning an off-the shelf 30 fps video camera into a variable depth of field 30 fps camera and a 30 fps range sensor.

focus to simultaneously obtain an all-focus video and a depth map video.

Traditional depth from focus/defocus approaches require the scene to be static during the acquisition of multiple images. In this chapter, we show how one can use optical flow information in order to remove this restriction and obtain 3D information in the presence of camera or scene motion. As a direct consequence of our ability to align multiple frames accurately, we are able to extract depth information and texture map at the native frame-rate of the camera, thereby converting a traditional 30 fps camera into a 30 fps extended depth-of-field (EDOF) camera and a 30 fps range sensor simultaneously.

2.1.1 Contributions

- DFD and EDOF imaging is extended to dynamic scenes by explicitly accounting for scene and camera motion. This enables a traditional video camera to acquire a 30 fps variable depth-of-field (DOF) video and act as a range sensor.

- The effects of (a) varying focal blur and (b) occlusions on the motion compensation problem are analyzed and an iterative refinement algorithm is designed that explicitly tackles such errors via reblurring.

- The design of a prototype is presented along with experimental results on several challenging scenes.

2.1.2 Prior Work

Depth From Defocus (DFD): Focus and defocus analysis for depth estimation has significant advantages over stereo and structure from motion as shown by [19, 20, 21], since they circumvent the correspondence problem. Another advantage of DFD over stereo is that only a single camera is required in DFD. Several algorithms for tackling DFD have been proposed [22, 23]. They minimize a cost function consisting of a data term and a spatial regularization term. The data term constrains how the texture blurs as a function of unknown depth and the known focal distances. The regularization terms model spatial smoothness constraints within the depth map of the scene, typically by penalizing the L_2 cost of the depth difference of neighboring pixels. However, none of these methods can handle the DFD problem in the presence of scene or camera motion. In this chapter, we present a framework to extend the DFD problem to scenarios with scene or camera motion.

Active Range Sensors: These sensors use an active pulse (e.g., laser, ultrasound) and either the direct time-of-flight or the phase difference between the emitted and received pulses to infer the scene depth. Structured light systems [24] use a pair of camera and projector to make the point correspondence problem in stereo easier. While the quality of the depth maps produced by such devices is usually high, they are expensive and require additional devices.

Variable Depth-of-Field (DOF) Imaging: The DOF of an imaging system can be extended by reducing the aperture. This however reduces the amount of light received by the camera, leading to low signal to noise ratio. On the other

hand, if we increase the aperture, the sensor noise is reduced but at the cost of the decrease in DOF. We would ideally want a large DOF but with reduced sensor noise. Recently, several approaches have been proposed [25, 26, 27, 28, 29] to overcome this fundamental trade-off between the sensor noise and DOF. Veeraraghavan *et al.*[26] and Levin *et al.*[27] use a broadband mask at the aperture making the point spread function of blur better behaved. This allows computational deblurring extending the DOF. Dowski and Cathey [25] increase the DOF by inserting a cubic phase plate near the lens, while Nagahara *et al.*[28] increase the DOF by moving the sensor during the exposure duration. In both cases the captured image is blurred, but the blur kernel is independent of depth and therefore can be deblurred using non-blind deblurring algorithms.

2.2 Basics and Limitations of DFD

A camera captures light from a scene and projects it on a sensor. Parts of the scene that are in focus are at depth s_0 given by the thin lens law:

$$\frac{1}{F_l} = \frac{1}{v} + \frac{1}{s_0}. \tag{2.1}$$

Here, F_l is the focal length of the lens and v is the distance between the camera lens and the sensor. Scene points that are at distance $s \neq s_0$ have a circle of confusion (blur) in the image plane. The distribution of light energy within this blur circle is referred to as the Point Spread Function (PSF). This PSF is a disc with its radius depending on the depth s of scene point:

$$\sigma = \frac{Dv}{2}\left(\frac{1}{F_l} - \frac{1}{v} - \frac{1}{s}\right), \tag{2.2}$$

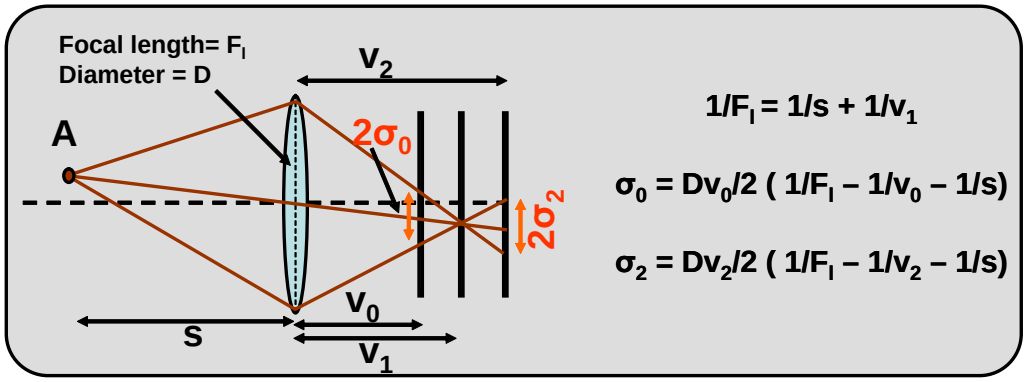


Figure 2.2: Depth from Defocus: The thin lens law states that when a scene point is out of focus, the blur radius of the out-of-focus scene point depends on the depth of the point apart from camera dependent parameters such as the focal length, aperture diameter and the sensor position. Given the camera parameters one can estimate depth from the blur radius.

where D is the lens aperture and σ is the radius of blur circle in the sensor plane. Figure 2.2 reviews this basic image formation in a camera. The scene point A with its depth s is in focus when the sensor is at distance v_1 from the lens. When the sensor is moved either towards the lens (v_0) or away from the lens (v_2), the rays from A form a circle of confusion with their radius given by equation 2.2. This dependence of blur on the depth of the scene point can be used as a cue to identify the depth of the scene, which is known as Depth from Defocus.

DFD in Static Scenes: Typical DFD methods capture a focal stack $\mathbb{F} = \{F_1, F_2, ...F_M\}$, consisting of a sequence of M images F_j captured at various focus settings. Consider the focal stack of a static scene shown in the top row of Figure 2.3. First (F_1) and last (F_5) images of a focal stack of 5 images for a static scene are

shown. In this static case, traditional DFD methods can produce an accurate depth map. The depth map can be used to obtain an extended depth-of-field (EDOF) image (also known as all-focus image) by combining the images in the focal stack.

Impact of Scene/Camera Motion: The basic assumption in traditional DFD methods is that the scene and camera are static. Consider the second row of Figure 2.3, where a focal stack is obtained as the red car moves to the right. The scene motion leads to correspondence errors in DFD, resulting in depth and texture errors. In EDOF images, the error appears as multiple copies of the moving object, while in the depth map, spurious depth edges are present on and around the moving object.

2.3 DFD in Dynamic Scenes

In this section, we describe how to adapt DFD methods to handle dynamic scenes. Let us assume that we have the motion information between a frame F_i and all the other frames within the focal stack $\mathbb{F}$. Intuitively, this information can be used to warp $\{F_j\}_{j=1}^{j=M}$ to F_i. This creates a "virtual" focal stack $\mathbb{Z}_i$ that corresponds to the time instant i and has the properties of a static focal stack. The virtual focal stack corresponding to F_3 is shown in the bottom row (Moving Car - Motion Warped) of Figure 2.3. Since the motion has been compensated for in this focal stack $\mathbb{Z}_3$, the scene points are in correspondence. Depth and texture maps for time instant i can then be obtained from this virtual focal stack.

The central problem in motion estimation is the presence of varying defocus

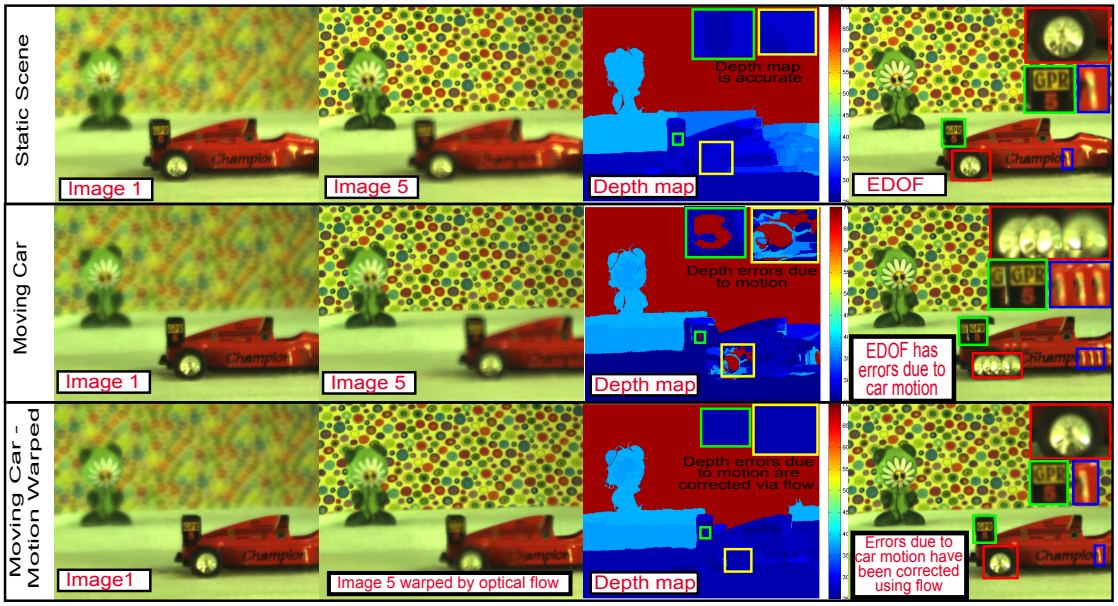


Figure 2.3: Effect of motion on DFD: Top row shows a static scene. A traditional DFD algorithm produces accurate depth and texture. Middle row shows a dynamic scene where the red car is moving to the right. This motion causes correspondence errors in DFD, resulting in depth and texture errors. Last row shows our algorithm applied to the scene in the middle row. Images shown here correspond to the virtual focal stack $\mathbb{Z}_3$ i.e., image 1 and image 5 have been warped to image 3 of the captured focal stack using optical flow. This allows the estimation of accurate depth and texture by correcting the correspondence errors. Depth values are in inches.

blur across the frames. Standard motion estimation techniques such as optical flow rely on the assumption of brightness constancy [30]:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \tag{2.3}$$

where (x, y) are the pixel grid points and t is the time instant. In Figure 2.4, we analyze the effect of varying defocus blur on flow estimation. The two frames are focused at different depths, and the pixel marked as A has different intensities. This is because varying focal setting blurs A with different kernels, leading to the violation of brightness constancy. This violation induces spurious optical flow as shown in the bottom row of Figure 2.4– Flow before reblurring. The magnitude of optical flow estimated in the static regions of the scene is around 2.5 pixels.

As shown in the example, in the absence of additional information, computing optical flow with changing blur levels is a challenging task. However, given the depth map and texture maps of the scene at time instant i, the flow can be solved accurately using reblurring (Figure 2.4– Flow after reblurring). Likewise, given the flow information the depth and the texture map can be recovered using DFD. This leads to an iterative refinement algorithm for estimating depth and texture of a dynamic scene via stage-wise optimization, which is presented in the next section.

2.4 Iterative Reconstruction of Depth and Flow

The overview of our reconstruction algorithm is given in Figure 2.5. Given the focal stack video, we initially compute the optical flow between the frames. We then warp the frames to generate a virtual focal stack for each time instant. The virtual

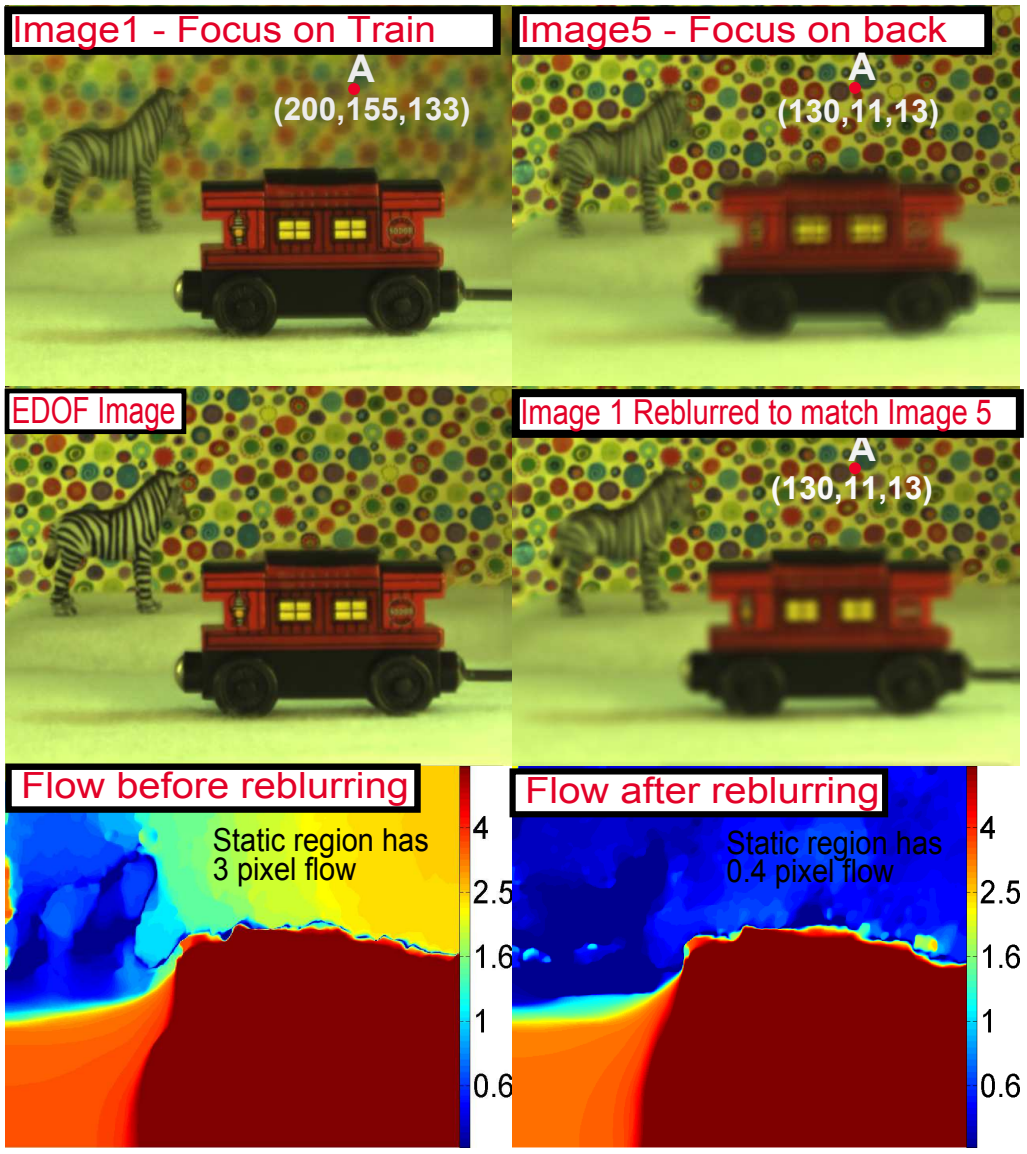


Figure 2.4: Flow estimation with varying blur. Top row shows two images of a focal stack with different blur levels. Observed intensities of pixel A violate the brightness constancy constraint for flow estimation. This violation introduces error in flow estimation. Texture and depth map (of image1) can be used to reblur image1 to match the blur level of image5. Third row shows the improvement in the estimated flow. This improvement is easily seen in static parts of the scene.

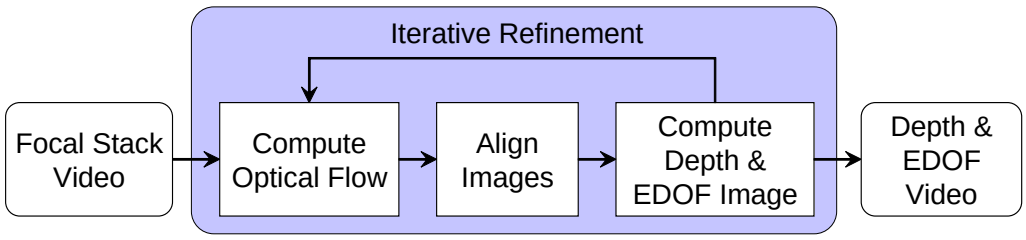


Figure 2.5: Block diagram of our reconstruction algorithm.

focal stack is used to compute depth and texture maps using DFD. The results are refined by alternating between flow estimation and DFD. The following subsections describe each step of the algorithm.

2.4.1 Optical Flow

In the first iteration, depth and texture maps are unavailable. Hence, flow estimation can be accomplished by first computing a coarse optical flow and then refining via reblurring during the subsequent iterations.

2.4.1.1 Initial Flow

Several approaches to compute the coarse initial flow can be adopted each with its own pros and cons. Spurious optical flow between frames i and $i+1$ could be computed in the first iteration and improved in subsequent iterations. This would result in errors caused by changes in blur as discussed earlier. Instead, we recover the initial flow by making an additional assumption — the flow has constant velocity within a period M of a focal stack. Frames from two consecutive focal stacks, (say) i^{th} and $(M+i)^{th}$ frames of a video, have the same blur levels, hence satisfy

brightness constancy. We compute the optical flow between i^{th} and $(M+i)^{th}$ frames and linearly interpolate the flow for the in-between frames. Although the initial flow is coarse due to the constant velocity assumption, it is refined via reblurring during the following iterations. Along with the constant velocity assumption, this approach also restricts the allowable motion between i^{th} and $(M+i)^{th}$ frames. In scenarios where this motion exceeds the allowable motion, the previous approach could be adopted.

2.4.1.2 Flow Given Depth and Texture

After the initial iteration of the DFD, the algorithm recovers a coarse estimation of depth and texture maps of the scene. Let D_i and T_i be depth map and texture map of the scene at time instant i respectively. This depth map D_i allows us to blur T_i with depth dependent kernels. This approach of matching the blur level of different images has earlier been utilized in shape from defocus [23, 31, 32]. Once the blur level of the two images are matched, brightness constancy is satisfied and hence optical flow can be computed with higher accuracy. Figure 2.4 shows the reblurred image F_5 and the computed optical flow between this reblurred image and F_1.

2.4.1.3 Occlusion

Here, we discuss the occlusion problem in motion compensation. Consider two frames F_1 and F_4 of a focal stack $\mathbb{F}$ shown in the top row of Figure 2.6. In this

dataset, red and blue markers rest on a turning table. When zoomed in, the red marker can be seen as moving to the right from F_1 to F_4. This motion occludes the pixels in the background region marked in blue. In F_1, this occluded region is blurred while it is occluded in other images of this focal stack. This implies that the information regarding the focused background for this occluded region is unavailable.

While warping F_4 to F_1 to compensate for the motion, this occluded region needs to be detected and filled in. The occluded region is detected by the inconsistency between forward and backward optical flows. The pixel is assumed to be occluded if forward-backward tracking results in a disparity greater than 2 pixels. The detected occlusion region is shown in the third row of Figure 2.6.

The information required to fill in the occluded region while warping F_4 to F_1, can be retrieved by looking at other frames with the same blur level i.e., F_{4+Mj} where M is the length of a focal stack and j is any integer. In other words, the values for this region can be identified by looking at other frames with same blur level which have pixels from the background in the occluded region. This is shown in the bottom right of Figure 2.6. However, this might not be possible when the occluded pixels itself are moving. For instance, in the case of camera motion, even pixels corresponding to the background move. In such cases, this region is filled up by copying corresponding pixels from the source frame, F_1 in this case which leads to visible artifacts on the object boundaries.

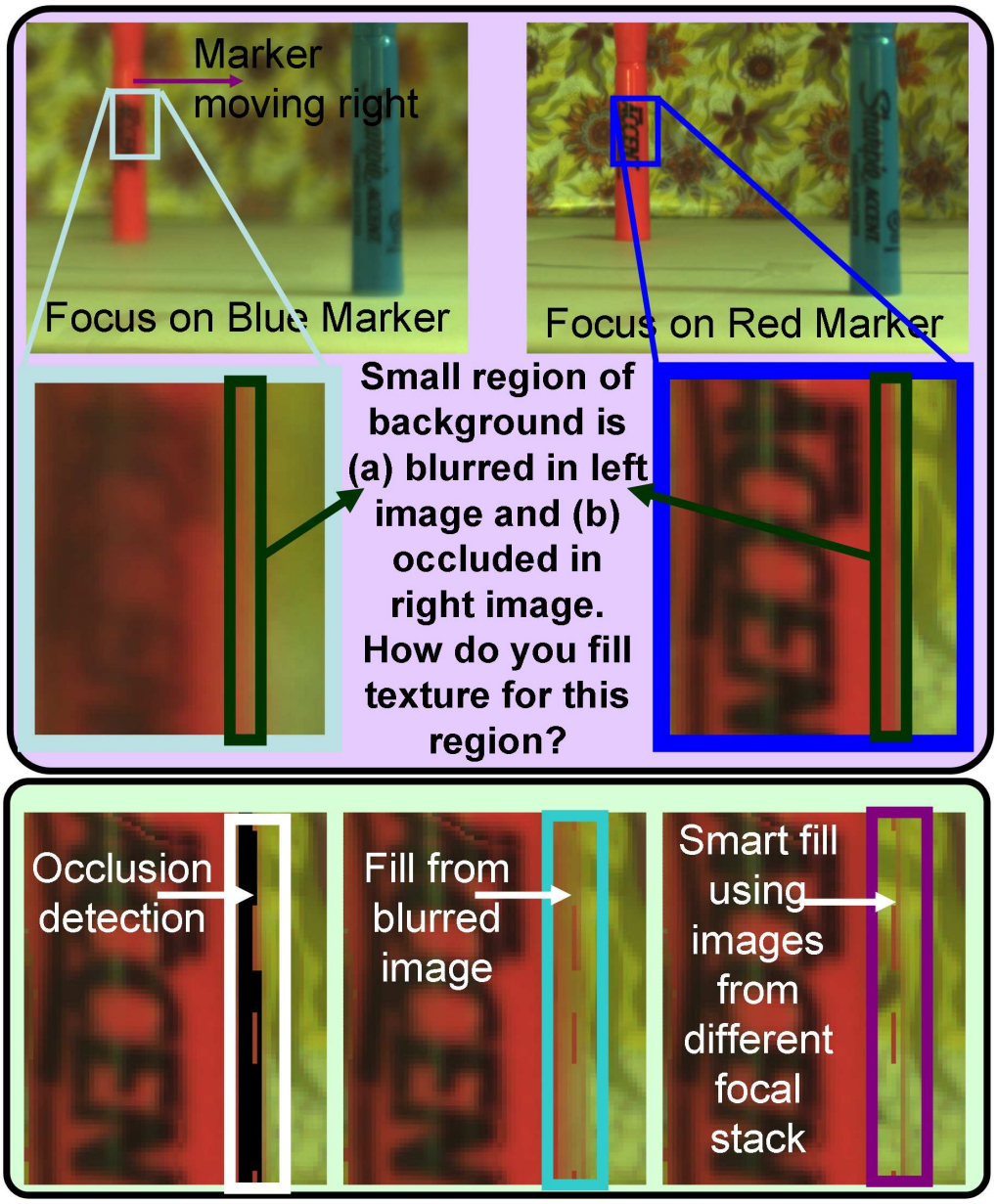


Figure 2.6: Handling occlusion. We detect occluded regions by checking the consistency between forward and backward optical flows between frames. If the occluded regions are static, we can fill them by using images from different focal stacks; otherwise we fill them using original (blurred) image in the current focal stack.

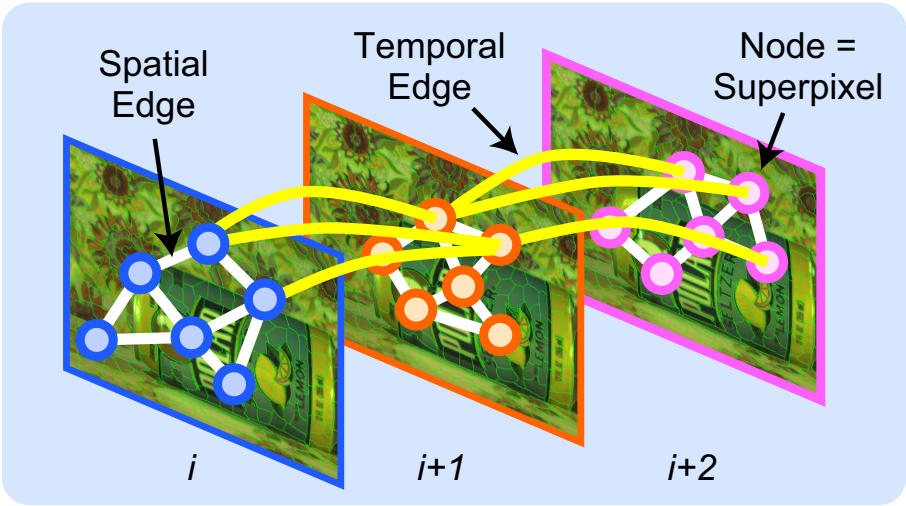


Figure 2.7: Pairwise MRF defined on superpixels. A spatial edge connects two nodes within an image if the superpixels share the boundary. A temporal edge connects two node across images if the superpixels have overlapped region after the alignment using optical flow.

2.4.2 Depth and Texture Given Optical Flow

After aligning the images in a focal stack using optical flow, we estimate depth maps $\mathbb{D} = \{D_1, D_2, ...D_M\}$ and textures $\mathbb{T} = \{T_1, T_2, ...T_M\}$ corresponding to each image F_i in the focal stack. We formulate the problem of depth estimation using a spatio-temporal Markov Random Field (MRF). As shown in Figure 2.7, we define an MRF using superpixels of the images as nodes and assume that each superpixel is represented by a front-parallel plane having a single depth value. These superpixels provide perceptually meaningful over-segmentation of an image which conserves most of the structure in terms of color and shape [33]. It provides advantages in terms of reduced complexity by moving from pixel grid to superpixel map and is also representationally efficient as it can model smoothness between all pixels in a superpixel. Similar to [34, 35], superpixel segmentation for each frame is obtained using an iterative algorithm where we initialize superpixels as a regular grid and update their shapes based on the current estimate of shape and color distribution of each segment. This produces superpixels which are regularly shaped.

Given a set of superpixels $\mathbb{P}$ and a finite set of depth labels $\mathbb{S}$, the objective is to assign a depth label $s \in \mathbb{S}$ to each $p \in \mathbb{P}$. The energy function of the MRF is represented as

$$E(s) = \sum_{p \in \mathbb{P}} D_p(s_p) + \alpha \sum_{\{p,q\}} V_{pq}(s_p, s_q), \tag{2.4}$$

where α controls the degree of regularization.

To compute the data term $D_p(s_p)$, we assume that textures T_i are available for each focal stack image F_i. Initially these textures are obtained by applying

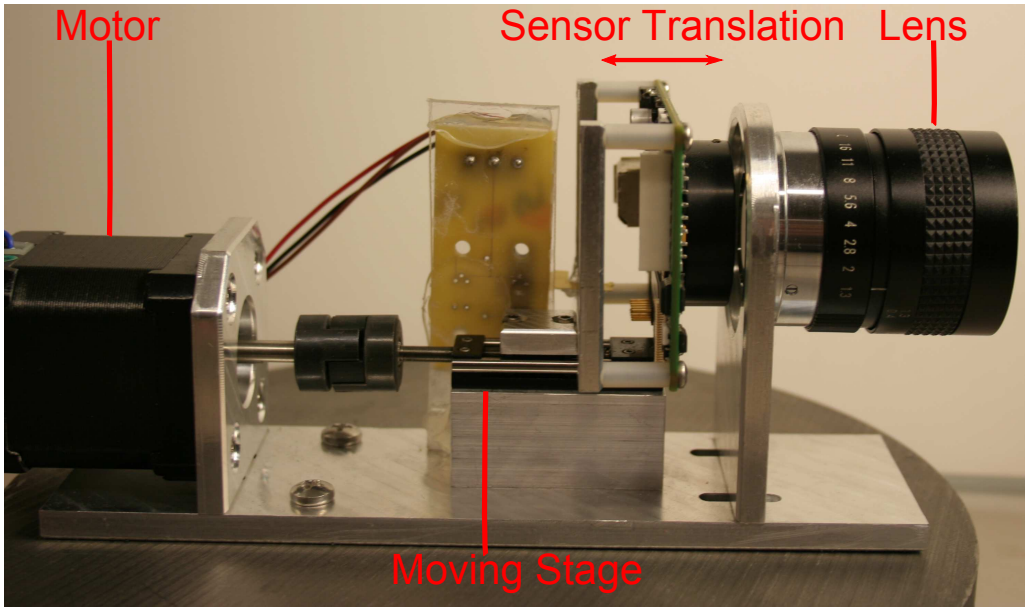


Figure 2.8: Our camera prototype.

photomontage algorithm [36] on the motion compensated focal stack $\mathbb{Z}_i$. After the first iteration we use textures computed using the previous estimate of the depth maps. Given the textures, the data term is computed by the sum of the squared difference between the observed superpixel and the reblurred superpixel for each depth level s_p. The PSF is assumed to be a disk kernel for reblurring the textures.

In our MRF formulation, we consider both spatial and temporal smoothness. Spatial smoothness penalizes changes in depth maps between two neighboring regions. We adopt the Potts model for spatial smoothness. Temporal smoothness, on the other hand, penalizes large change in depth between two consecutive frames. The smoothness term is computed as

$$V_{pq}(s_p, s_q) = w_{pq}\, T(s_p \neq s_q), \tag{2.5}$$

here, $T(.)$ is the indicator function with value 1 if its argument is true and 0 if it is

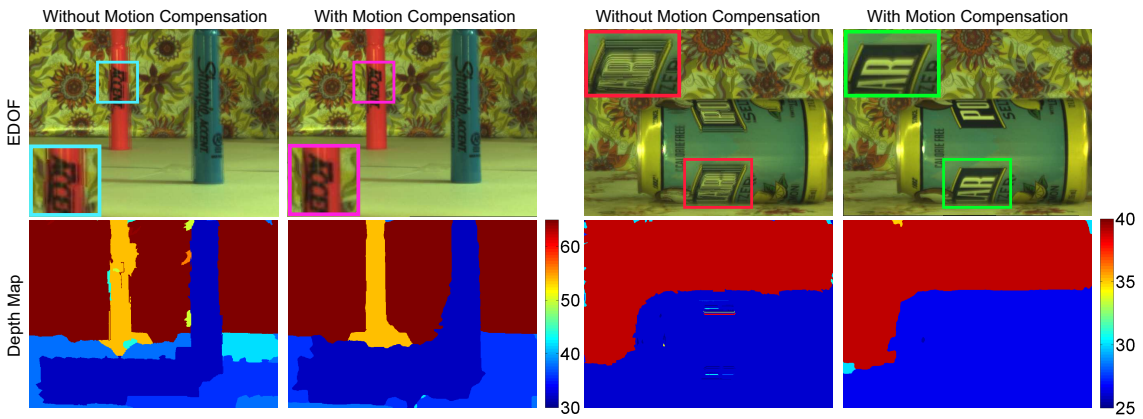


Figure 2.9: Comparison between EDOF images and depth maps generated with or without motion compensation. Traditional DFD without motion compensation fails to handle moving objects and produces artifacts at boundary regions. In the 'rolling can' example on the right, depth map artifacts can be seen corresponding to the texture map artifact highlighted for the result without motion compensation. Our algorithm correctly compensates for the object motion and generates high-quality depth maps and EDOF images. Depth values are in inches.

false. This term penalizes depth discontinuities between the neighboring superpixels p and q, and w_{pq} is a spatial/temporal weighting factor. The weight w_{pq} between two spatially neighboring superpixels p and q is determined by the similarity in the average color of the two superpixels: $w_{pq} = \exp\left(-\frac{||I_p - I_q||^2}{\tau}\right)$. Weights for the temporally neighboring superpixels are determined in the following way: Consider frames A and B. Let u be the optical flow between these two frames. Superpixel $p \in A$ is warped to frame B using the optical flow u. The overlap of p with the superpixels of frame B is then used as weights between temporal neighbors. We use graph cut algorithm [37] to minimize the energy function.

2.5 Experiments

In this section we discuss our prototype design and results on few challenging scenes.

2.5.1 Camera Prototype

Figure 2.8 shows our camera prototype. It translates a 1/3 inch Sony progressive scan CCD with a stepper motor which drives the National Aperture Inc., MicroMini Stage $MM-3-X$. The sensor moves around $2\mu m$ in each motor step. Thus, the distance between two consecutive sensor positions to capture the image can be varied at multiples of $2\mu m$. This can be controlled according to the DOF of the scene to be captured and the number of images required per stack. In our experiments, we used 67 steps ($= 134\mu m$) between two consecutive sensor positions. A C-mount lens with a focal length of $12.5mm$ is attached to the fixed optical stage. While capturing the video, we move the sensor continuously and typically keep our exposure time to be $10ms$. A very small translation of the camera sensor covers a large amount of focal depths [28].

The camera translates along the optical axis of the lens with a constant speed. When it reaches the pre-specified extremum in one direction, the camera translates in the opposite direction. Hence it is capable of continuously capturing images at 30 fps. In most of our experiments, during one half period of sensor motion, the camera captures 5 images, which is the size of the focal stack.

Calibration: The magnification of the imaging system changes due to the

translation of the sensor. In addition, the direction of the translation is not perfectly aligned with the central axis of the camera. These effects induce a planar projective transformation between the focal stack images. During calibration, we place a calibration grid in the scene and estimate the planar homographies between the the focal stack images. These transformations are used to warp the images and processing is done in a canonical space (coordinate frame of F_3).

The relation between the scene depth and the blur kernel size is necessary for reblurring. During calibration, we place the calibration grid at multiple depths. At each depth, the focus is on the grid in one of the focal stack images, and the blur kernel sizes for all the other focal stack images are computed. The blur kernel size for a given depth is then estimated via linear interpolation.

2.5.2 Results

DFD/EDOF for Dynamic Scenes: We collected several challenging datasets of dynamic scenes with our prototype. Shown in Figure 2.9 are the results on two different datasets where we compare the DFD and EDOF images before and after explicit motion compensation using our algorithm. The dataset on the left consists of two markers, with the orange marker moving to the right while the blue marker is moving to the left. The dataset on the right consists of a can rolling towards the camera. The first and third columns show the EDOF images and the depth maps before motion compensation. Notice the errors in the depth map due to object motion. The second and fourth column shows the EDOF images and depth map after motion compensation. Notice the significant increase in depth accuracy

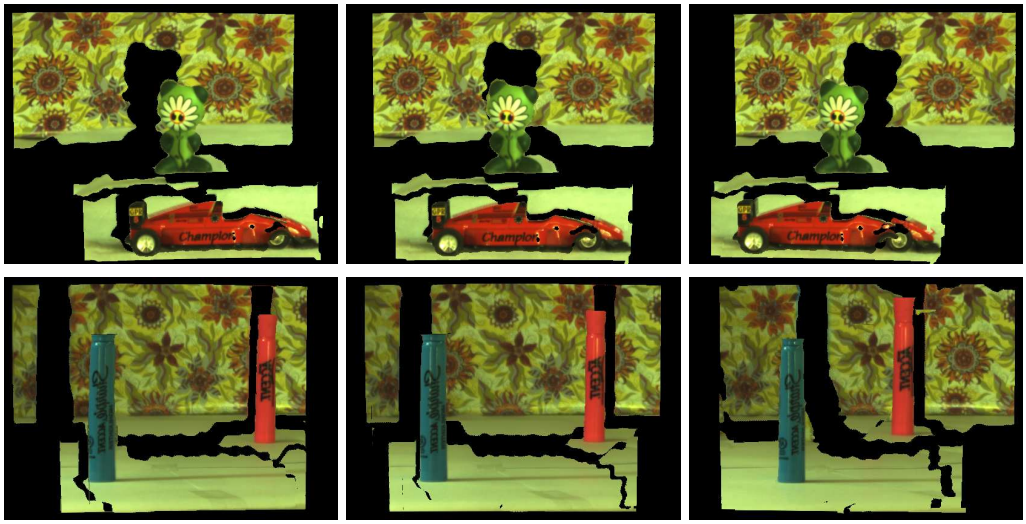


Figure 2.10: Novel views rendered using the depth and texture maps from different viewpoints.

in the marker dataset and the reduction of motion artifacts in the EDOF images of the can dataset. This estimated depth and texture maps enable us to render novel views. Figure 2.10 shows frames from marker dataset and the toy dataset rendered from 3 novel views. Accuracy of the retrieved boundaries for the objects can easily be visualized in these novel views. As the bottom surface is texture-less, no depth information is recovered on the surface.

Figure 2.11 shows a dataset of desktop where the coffee mug is being picked up. This dataset has 4 captured images in each focal stack. Two images from this dataset are shown in the top row of the Figure. The left column corresponds to the near-focus image while the right column corresponds to the far-focus image. EDOF images and depth maps are shown in the second and third rows respectively. It can be noted that EDOF images have all parts of the image as sharp. Depth map

recovered for the textured regions are close to the ground truth values while few errors can be observed at the texture-less regions.

Effect of Refinement: Figure 2.12 shows the importance of the iterative optimization described in Section 2.4. This dataset consists of a toy scene in which the red car is moving to the right. Notice that both the depth map and the EDOF images have flow artifacts after the initial iteration (constant velocity flow assumption); whereas these artifacts are significantly reduced after second iteration via reblurring, leading to depth and texture maps as shown in the right column of Figure 2.12.

Reduced Depth of Field: Since our method computes both depth and texture simultaneously for dynamic scenes, this enables us to synthetically reduce the depth of field (DOF) of the acquired images. The depth quantization of the DFD algorithm is much finer than the DOF of each of the captured focal stack images. Shown in Figure 2.13 is a dataset including a book on magic. This book was translated to the left and towards the camera. Shown in the first row are two of the captured images showing the translation and the DOF of the captured images. The corresponding EDOF images estimated by our algorithm are shown in the second row. The depth map at time t_1 is shown in the left of third row. Finally, the depth map and the EDOF image are used in a depth dependent reblurring framework to produce the reduced DOF image shown in the right of third row. Notice that the DOF of this rendering is about 3 times smaller than the original DOF of the captured focal stack. Thus our algorithm can be used either to extend or reduce the DOF. Reducing the DOF is important for inexpensive and small cameras (like

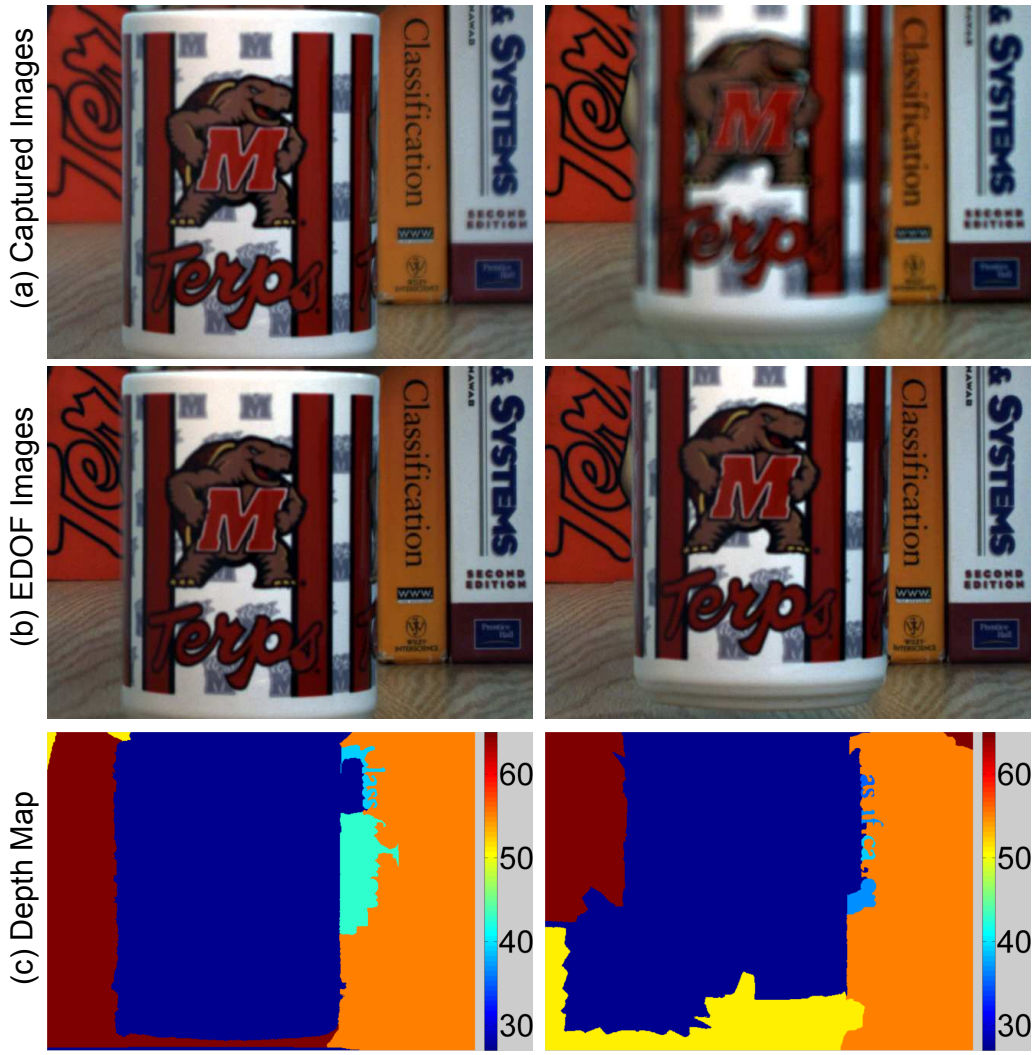


Figure 2.11: (a) Two images from a desktop scene with coffee mug being picked up with left one near-focused and right one far focused. (b) Corresponding EDOF images. (c) Corresponding Depth maps. Depth values shown here are in inches.

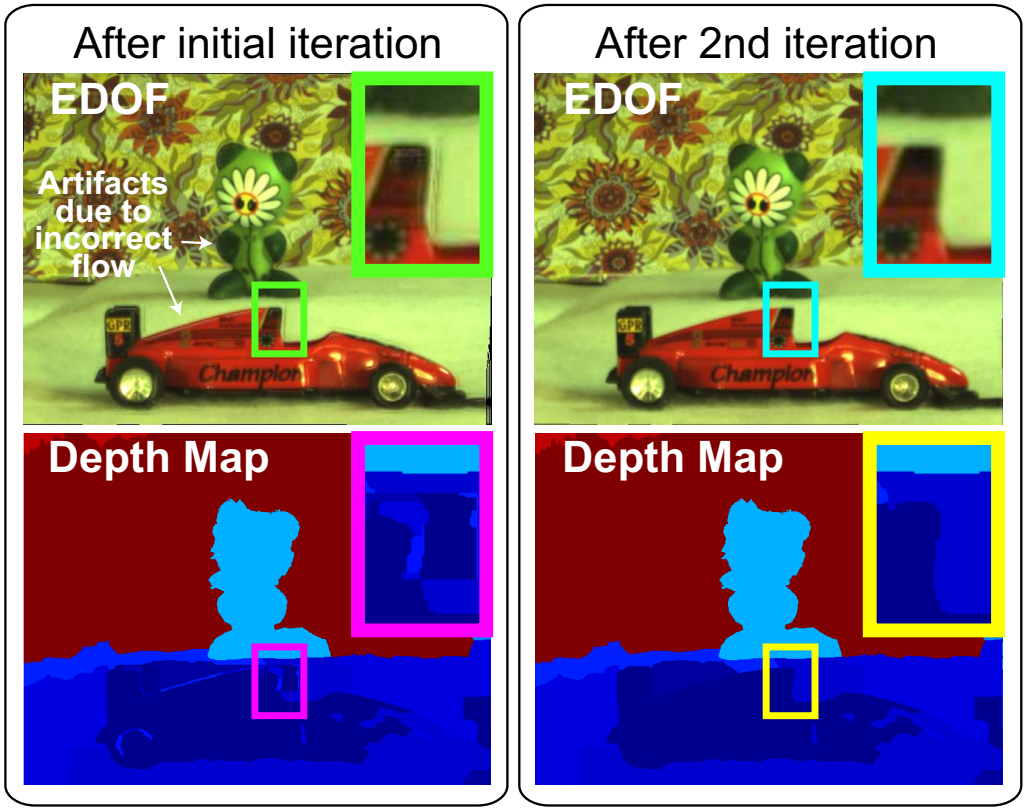


Figure 2.12: EDOF images and depth maps (left) after initial iteration and (right) after second iteration. Zoom-ups from each image has been shown. Note the boundary artifacts are reduced after the second iteration.

cellphone cameras) that have a very large DOF because of their small aperture size.

2.6 Conclusion

In this chapter, we have shown how advances in optical flow and motion estimation can be exploited towards extending the problem of DFD and EDOF imaging to dynamic scenes. The idea of performing explicit motion compensation is general and can be used with any of the available state-of-the-art DFD algorithms. Though in this chapter, our implementation is based on a graph cut formulation, the ideas proposed here can be easily extended for other DFD approaches. Our analysis shows the importance of accurate flow estimation, describes some of the fundamental challenges in this problem due to focal blur,and suggests methods to handle these challenges.

2.6.0.1 Limitations

Our method suffers from the disadvantages of traditional DFD, such as the need for scene texture and a lens with large enough aperture to cause depth of field effects. It further inherits the limitation of dense motion estimation algorithms i.e., motion between successive frames of the video should not exceed the limitations of optical flow estimation algorithms, and motion component parallel to the axis of the camera cannot be estimated reliably. Further, the depth range in the scene should be small enough so that (a) the number of images in a focal stack is kept reasonable, and (b) the maximum blur of any observed scene point is manageable. The choice

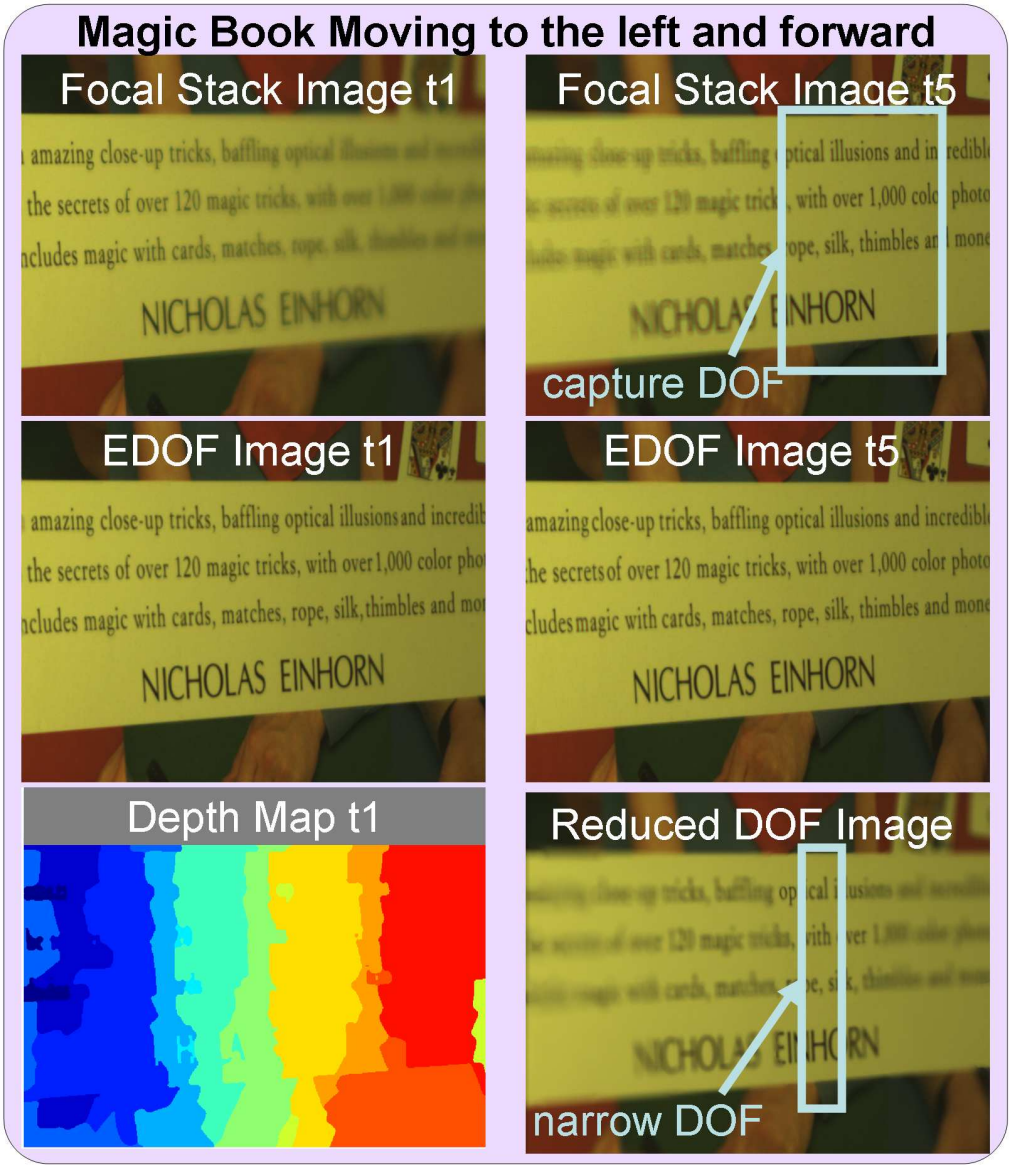


Figure 2.13: Variable depth-of-field (DOF) imaging. The estimated depth map enables us to compute extended DOF images and reduced DOF images.

of the initialization scheme for the optical flow limit the maximum motion possible between two images with the same focal depth. This will be addressed in future work where initial optical flow could be solved by estimating motion for multiple frames together while compensating for blur in the same framework.

The main advantage of our approach is that this turns video cameras into high resolution variable depth of field and range sensors. Since most video cameras today have a built-in auto-focus feature that allows us to vary the focal distance during video acquisition, our approach could convert an off-the-shelf video camera into a range sensor also. Unfortunately, the firmwares available with cameras today do not provide external access to the auto-focus mechanism. The FrankenCamera [38] is an indication that the field is poised to change this lack of ability to control the hardware features available on the camera platform. We hope that this and several similar research directions [28, 39, 40] have the positive effect of convincing camera and lens manufacturers to allow external access to some of the in-built functionalities. Such external access, we believe, would turn traditional video cameras into powerful devices that are capable of improving our ability to do image understanding from the acquired videos.

Chapter 3

Bin Picking of Specular Objects

3.1 Introduction

Steady advances in machine vision algorithms have resulted in the adoption of these techniques in industrial automation and assembly. Nevertheless, several challenges still persist in tackling objects that do not have a near-Lambertian reflectance. Objects with mirror-like, transparent or translucent surfaces possess material properties that are currently viewed as noise sources and traditional techniques attempt to suppress the impact of these effects. This means that objects which are either highly specular or have significant translucency cannot be handled by such methods since these material effects cannot be completely suppressed.

In this chapter, we show that the effect of specularities need not be treated as a source of noise but rather as a feature that can and does enhance our ability to detect, recognize and interact with specular objects. We show that regions of high curvature on a specular object lead to a very consistent and robust feature that can be used to reliably perform machine vision tasks. The basic idea is very simple. Since high curvature regions in an object have normals that span a wide angular extent, these regions almost always produce specular reflections irrespective of the lighting position. The detection of these specular reflections serves to locate regions

of very high curvature on the surface of the object. The use of a multi-flash camera (MFC) [41] aids in the detection and extraction of these specular features. These high curvature regions provide a distinct signature for several industrial objects, which can be used for object detection, recognition and pose estimation.

In this chapter, we primarily focus on demonstrating our approach to grasp screws from a cluttered bin, as shown in Figure 3.1. Arguably, screws form the most fundamental class of objects used in manufacturing systems. More threaded screws are produced each year than any other machine elements [42]. In conventional assembly lines, the screws have to be placed into a part holder with known position and orientation before the robot grasps and manipulates them. This operation requires either a specially designed hardware for each screw type such as a part feeder or performed using manual labor, which can be avoided by our bin picking system.

Majority of screws are made from shiny materials; therefore they cannot be handled easily by traditional machine vision algorithms. In addition, pose estimation of a screw in a bin is a very challenging problem because of clutter and occlusion. We show that the specular features can be robustly extracted even in a cluttered bin. We estimate the pose of screws by matching these features from the same screw in multiple views. This matching is particularly difficult because the entire bin contains many instances of the same screw in a wide variety of poses. To solve this, we employ three-plane rank two constraint for establishing correspondences and two/three-view triangulation for pose estimation.

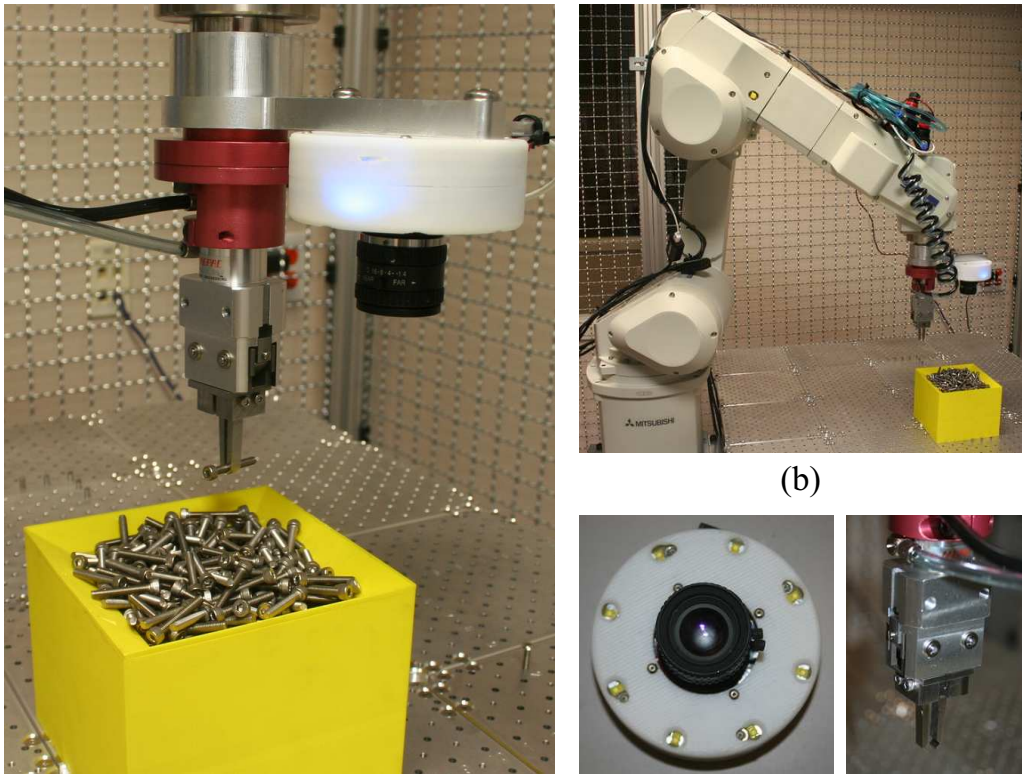


Figure 3.1: **The robotic grasping platform.** (a) Our system enables the grasping of screws from a cluttered bin using a multi-flash camera (MFC). (b) System overview. MFC is mounted on the robotic arm. Screws are placed in a part container. (c) Close up of MFC, employing 8 LEDs uniformly placed on its circumference. (d) Close up of the gripper.

3.1.1 Prior Work

Vision-based bin picking has attracted a lot of interest in the last decade [43, 44, 45] where the main problem is to localize an object and estimate its pose. Development of such systems has been challenging mainly because of (a) the specular reflections from the metallic surfaces of industrial parts and (b) occlusions in a cluttered bin.

Model-based pose estimation algorithms using 3D model to 2D image correspondences can be found in [46, 47, 48]. Unfortunately, 3D-2D point correspondences are hard to obtain for industrial parts due to their textureless surfaces. The situation is particularly severe when multiple identical objects are placed together and overlap each other.

Object contours provide rich information about object identities and their poses. Various contour matching algorithms have been proposed in [49, 50, 51]. However, for specular objects the contour information is difficult to obtain in a cluttered bin since these objects do not have appearance of their own but reflect the surrounding environment.

Range sensors have been widely used for the pose estimation problem. Bolles and Horaud [52] use range data to group the surface features which is then used to generate and verify the hypotheses of object location. Wang et al. [53] use 3D range data to compute shape of flexible industrial parts such as cables. However in the presence of specularities, range sensors fail to produce accurate depth maps and they are comparably more expensive than camera-based solutions.

Active illumination patterns can greatly assist vision algorithms for extracting robust features. Horn et al. [43] use brightness of patches observed with varying illumination condition to estimate the orientation of surface patches and then match them with the 3D model. In our approach, we exploit MFC [41] to extract specular features in the regions of high curvature. MFC has 8 LEDs that are uniformly placed on its circumference and flash one by one. As the illumination sources move, the specular reflections also move by a distance that is inversely proportional to the surface curvature.

Specularities have generally been treated as sources of noise in machine vision algorithms. Most vision algorithms identify specularities and remove them before inference. Brelstaff and Blake [54] identify regions with specular highlights which deviate from Lambertian reflectance. Nayar et al. [55] use polarization filters to identify and discard specular highlights. Mallick et al. [56] transform the color space such that it becomes invariant to changes due to specular highlights.

Robust feature extraction in the presence of strong specularities has always been a challenging task. Specular highlights have been used in vision and robotics for object detection and pose estimation [57]. Healey and Binford [58] use specular highlights to reconstruct the local orientation and principal curvatures of a surface. Similarly, Gremban and Ikeuchi [59] use specular highlights for object recognition, and plan novel views that are discriminative between objects with similar highlights.

Sankaranarayanan et al. [60] propose a novel image invariant for highly specular and mirror-like surfaces exploiting the fact that the image gradients exhibit degeneracy at regions where at least one principal curvature is zero. This property

is used to detect points with parabolic curvature on the object surface. However, this is not a very practical feature for industrial objects, since parabolic curvature points rarely exist on the surface of such objects. In this chapter, we consider the exact opposite: i.e., the regions on the surface of the object that have very high curvature. Such regions have a large variety of normals present within a small area and therefore produce specular reflections irrespective of the illumination position.

3.1.2 Contributions

The main technical contributions of this chapter are

- We show that regions of high curvature in highly specular objects produce specular reflections largely independent of the location of the illumination sources, which can be used as a distinct features for object detection and pose estimation.

- We show that an inexpensive multi-flash camera can be used to reliably detect such specular features.

- We use screws as an example part and develop two view and three view-based algorithms for 3D pose estimation using triangulation.

- Finally, we implement the algorithm on an industrial robot and show very high location and angular accuracy for 3D pose estimation.

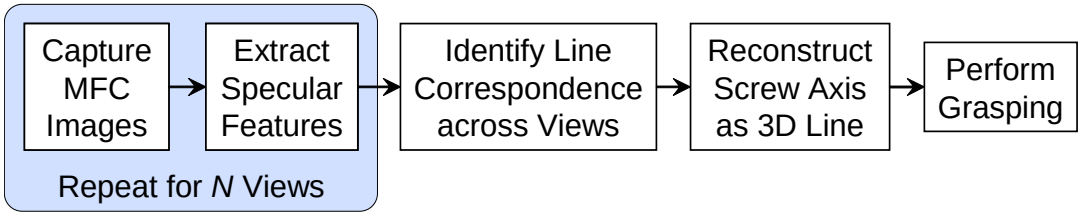


Figure 3.2: **Overview of our algorithm.**

3.2 System Overview

Figure 3.1 shows our system performing bin picking of screws. We use a 6-axis industrial robot arm, on which the MFC is mounted. The MFC was calibrated both internally and externally using a checker board. Hand-eye (gripper-camera) calibration was also performed so that the robot arm can interact and grasp objects using the gripper.

Figure 3.2 shows the overview of our algorithm. We employ a multi-view approach to compute the pose of screws in a bin. We first capture a set of images using the MFC and extract specular features, as we describe in Section 3.3. We repeat these processes for two or three capture positions. We then identify the lines corresponding to the same screw across multiple views and estimate the pose of a screw by reconstructing the 3D axis of the screw, as described in Section 3.4. Our system finally grasps the screw using the estimated pose and performs the subsequent assembly task.

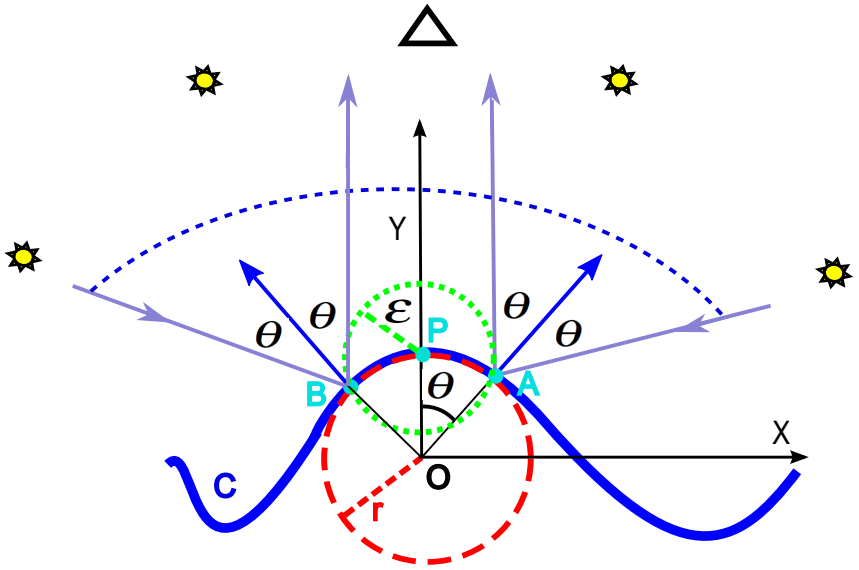


Figure 3.3: **Analysis of the relation between curvature and the cone of rays.** Point P has curvature κ. If the illumination direction is within the cone (shown by blue dotted curve) of $[-2\theta \;\; 2\theta]$, then the specular highlight is captured by the camera within the ϵ neighborhood (shown in green).

3.3 Specular Feature Extraction

In this section, we analyze the reflection of light rays from the surface of a specular object and derive a feature extraction algorithm for detecting high curvature regions.

3.3.1 Reflection Analysis

Consider a point P on an arbitrary one dimensional curve C as shown in Figure 3.3. Let r be the radius of the osculating circle at P. Then the curvature κ at P is given by

$$\kappa = \frac{1}{r}. \tag{3.1}$$

Consider an ϵ neighborhood at point P as shown in Figure 3.3. As ϵ is small enough, we assume the curvature in this neighborhood to be constant for our analysis. Without loss of generality, we consider the two-dimensional coordinate axes with origin as the center of the osculating circle O, Y-axis passing through P and X-axis orthogonal to it. This ϵ ball meets the curve C at A and B with θ given by

$$\theta = 2\sin^{-1}\left(\frac{\epsilon\kappa}{2}\right). \tag{3.2}$$

Now, consider the ray towards A from a camera placed on Y-axis. The camera center is assumed to be at $(0, y_0)$ such that $y_0 \gg \epsilon$. This implies that the ray coming from camera center can be considered parallel to the Y-axis as shown in the figure. This ray subtends an angle θ with the normal at A and gets reflected at angle θ from the normal. Symmetrical analysis holds true for point B. This shows that if the light source is placed anywhere within this cone (shown by blue dotted curve) of $[-2\theta \;\; 2\theta]$, then the camera will receive specular reflection from this ϵ neighborhood.

For a fixed cone $[-2\theta \;\; 2\theta]$, the size of the ϵ neighborhood is inversely proportional to the curvature of the point,

$$\epsilon = \frac{2}{\kappa}\sin\frac{\theta}{2}$$

by using the distant camera assumption. As the curvature increases, the reflections are visible within a small neighborhood of the point. In contrast, when the curve is almost flat (curvature is close to zero), the reflection is not visible within the neighborhood of the point. In the next section we illustrate methods that exploit this fact to detect features points on the high curvature regions of the object. This analysis assumed mirror-like reflection. When the specular lobe is considered, this

cone of rays can be increased by an additional 2σ where σ is the width of specular lobe.

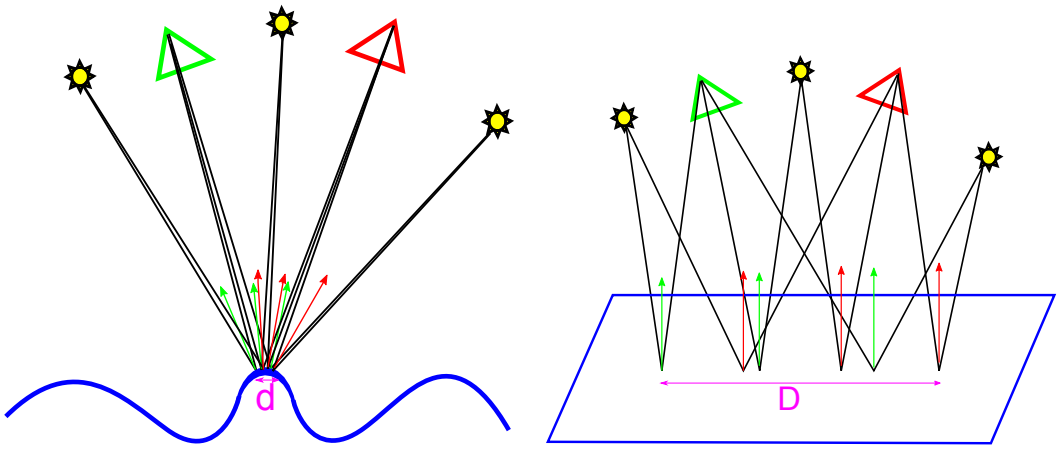
The analysis can be extended for a two dimensional surface S. The principal curvatures are defined as the minimum and maximum values of the curvature measured along various directions at a given point. The Gaussian curvature K of a surface is given by the product of principal curvatures κ_1 and κ_2 of the point

$$K = \kappa_1 \kappa_2. \tag{3.3}$$

Similarly for a two dimensional surface, the reflections are visible within a smaller neighborhood of the point as the Gaussian curvature of the point increases. Note that both principle curvatures have to be large to observe the reflection within a small neighborhood. For example, a sphere (with small radius) is a surface with both its principal curvatures large ($\kappa_1 = \kappa_2 = 1/r$ for all the points on the sphere), therefore the reflection is visible within a small neighborhood. In contrast, the Gaussian curvature of a point on a cylindrical object is zero since the surface has bending only in one direction. Hence the reflection may not be visible within an immediate neighborhood.

3.3.2 MFC-Based Feature Extraction Algorithm

Our feature detection algorithm finds points on the specular surface which have *large Gaussian curvature* (which has bending in both directions) and *normals towards the camera*. Although the second requirement seems like a restriction, in fact the high curvature regions span a large set of surface normals. These features



(a) (b)

Figure 3.4: (a) High curvature region. The specular highlights shift by a small amount with the changing light location. (b) Low curvature region. The specular highlights shift by a large amount.

provide sufficient information for 3D reconstruction and pose estimation which are explained in the Section 3.4.

An MFC is an active illumination based camera that contains 8 point light sources (LEDs) arranged in a circle around the camera as shown in Figure 3.1. As the different LEDs around the camera flash, the specular highlights on the surface move depending upon the local curvature. From the analysis presented in the previous section, it is clear that the specular highlights remain in a very local neighborhood for all points that have high surface curvature. We exploit this cue to detect pixels that correspond to the high curvature regions on the object having normals towards the camera. These pixels serve as a feature that is both characteristic of the object's 3D shape and its relative pose with respect to the camera.

Our specular feature detection algorithm consists of the following steps. We

first take 8 images corresponding to 8 flashes, along with a 9^{th} image with no flash. This image with no flash (ambient image) is then subtracted from each image to remove the effects of ambient illumination. Let us denote the image captured (after ambient subtraction) during the flashing time of the i^{th} LED as I_i. The minimum intensity values at each pixel location among these ambient subtracted images are found and used to construct the minimum illumination image

$$I_{min}(x, y) = \min_i I_i(x, y)$$

. The minimum illumination image will be similar to the surface albedo. Since the surfaces we are considering are highly specular and the specular highlights move across the images, the minimum illumination image appears to be dark for all the specular regions. We compute the ratio images of the ambient subtracted images to the minimum illumination image,

$$RI_i = \frac{I_i}{I_{min}}$$

. Ideally, the ratio values in the non-highlight regions remain close to one while the ratio values in regions seeing a specular highlight are much greater than 1. This information is utilized to detect the highlight regions at each flash image.

As discussed above, with the changing flash position specular highlights at high curvature regions stay within the small neighborhood (Figure 3.4(a)) whereas they shift by a large amount in low curvature regions (Figure 3.4(b)). This effect is explained in Figure 3.5.

We choose a neighborhood ϵ (set to 1 pixel in our experiments) and compute the number of flash images in which a specular highlight was observed within this

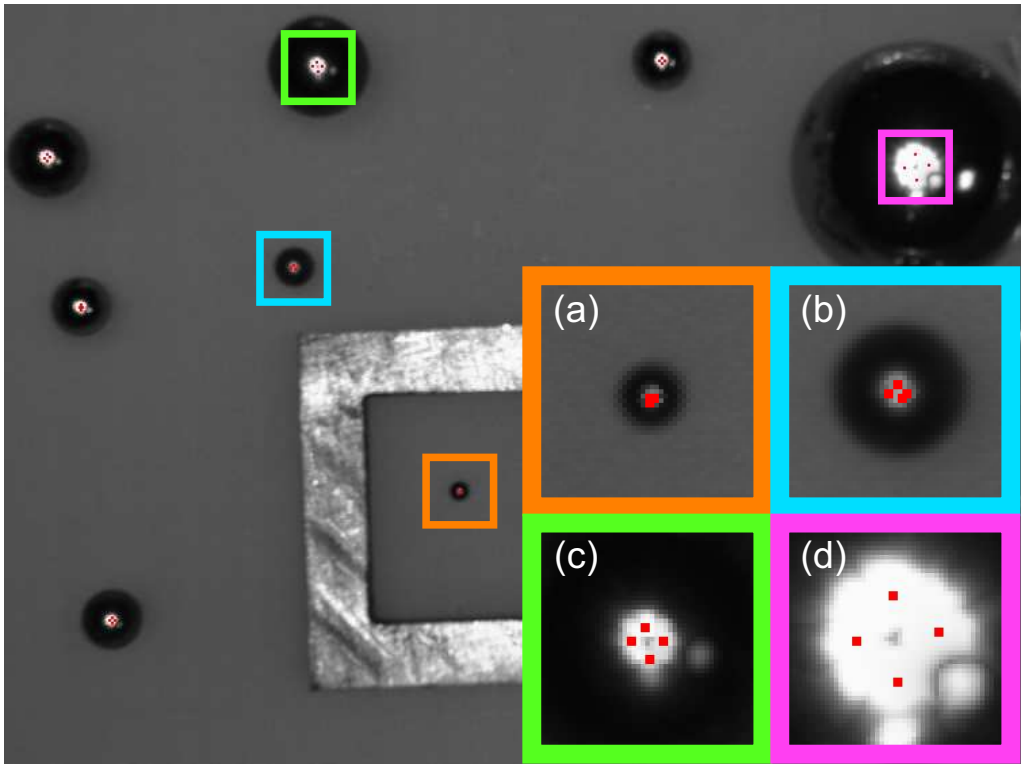
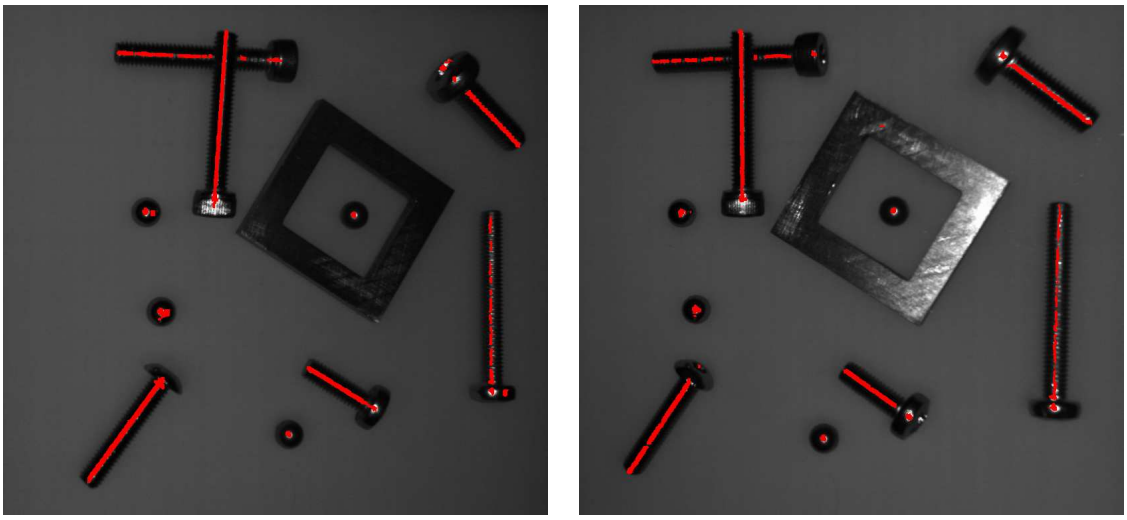


Figure 3.5: **Motion of specular reflections for different sizes of spheres.** The centers of specular highlights corresponding to 4 different light positions are superimposed as red dots on the maximum image. Close-ups of 4 spheres are shown in bottom right. The diameter of sphere and the diameter of the motion of specular highlight are respectively (a) 1.2 mm–2 pixels, (b) 3.2 mm–4 pixels, (c) 8 mm–7 pixels, and (d) 19 mm–18 pixels. The smaller the diameter (the larger the curvature), the smaller the motion of specular highlight.



(a) View 1 (b) View 2

Figure 3.6: **Specular feature detection using MFC.** The high curvature regions facing towards the camera are robustly identified across two views whereas low curvature regions and Lambertian surfaces are filtered out.

ϵ neighborhood. For pixels corresponding to high curvature regions, the specular highlight remains within this ϵ neighborhood and therefore a specular highlight is observed in all 8 MFC images within this ϵ neighborhood. For pixels corresponding to low curvature regions, the specular highlight moves outside the ϵ neighborhood and therefore the number of images in which the specular highlight is observed within the ϵ neighborhood is less than 8. Pixels that correspond to Lambertian surfaces are filtered out automatically because of normalization with respect to the minimum image. An example of specular feature detection using MFC is shown in Figure 3.6. The algorithm robustly identifies high curvature specular regions facing towards camera across two views while filtering out Lambertian surfaces and low curvature specular regions.

3.4 Pose Estimation of Screws

This section presents a practical application of the specular features described in Section 3.3, in conjunction with a novel pose estimation algorithm for bin picking of screws.

3.4.1 Specular Features on Screws

Screws are cylindrical in shape. As discussed in Section 3.3, curvature of cylinder in direction perpendicular to the axis of the cylinder is high but in the direction parallel to the axis is small. This renders the Gaussian curvature of cylinder to be small. Let us now consider the effect of adding threads on the surface of screws. The threads (typically approximated as a helix or a conical spiral) provide high curvature to all the points on the screw body even in the direction parallel to the axis of the cylinder. This, in turn, ensures that the specular highlights are visible in an ϵ neighborhood independent of the illumination direction.

The specular features are extracted using the algorithm described in Section 3.3. It is important to analyze the location of the features detected on the screw. The specular features are detected on high curvature regions having normals towards the camera. Approximating the screw shape with a helix, the points on the helix facing towards the camera lie on a line. In addition, this line lies on the plane joining the camera center and the axis of the screw as shown in Figure 3.7.

We therefore represent the specular features on the screws with line segments. For line fitting we use a variant of RANSAC [61] algorithm. The algorithm initially

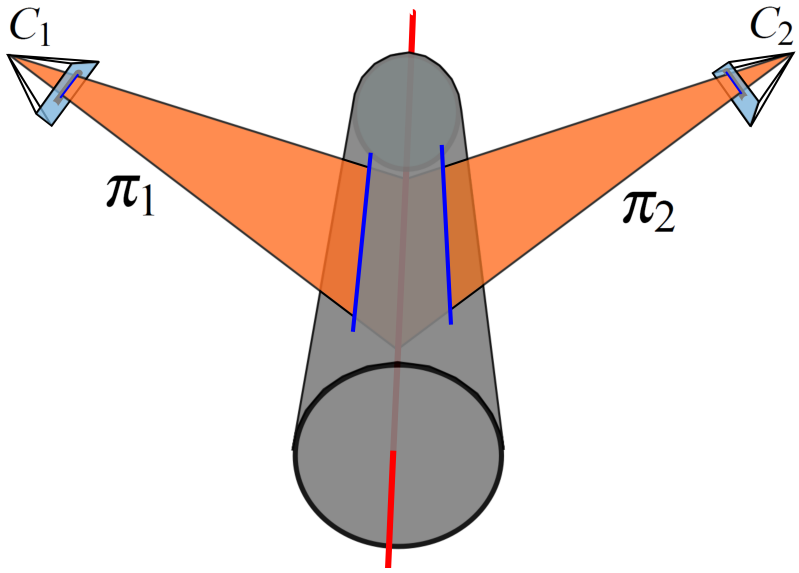
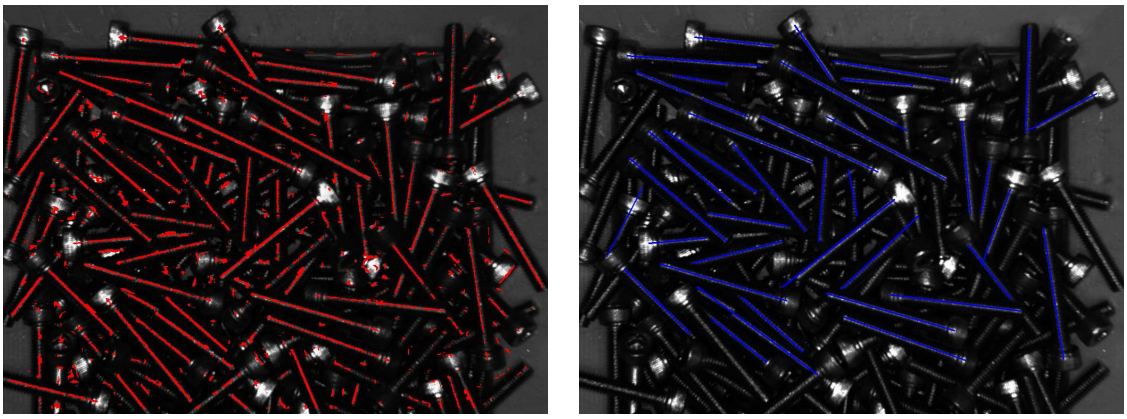


Figure 3.7: **Central axis reconstruction.** For simplicity the screw shape is shown as a cylinder. Blue lines shown on the surface of the cylinder represent the specular features detected. π_i represents the planes formed by these lines with the respective camera centers. π_1 and π_2 intersect at the axis of the cylinder (shown in red).



(a) (b)

Figure 3.8: (a) Detected specular features on screws. (b) Fitted lines.

hypothesizes a variety of lines by selecting a small subset of points and their directions. The support of a line is given by the set of points which satisfy the line equation within a small residual and form a continuous structure. The line segment with the largest support is retained and the procedure is iterated with the reduced set until the support becomes smaller than a few points. The RANSAC algorithm provides inlier points for each line segment. We then refine each line segment using least square estimation on the inlier points. In Figure 3.8, we show the detected specular points from a bin of screws and their line segment representation.

3.4.2 Pose Estimation

In order to estimate the pose of a screw, we reconstruct the 3D line corresponding to the axis of the screw which uniquely determines the orientation and the position of the screw. We compute the specular features from multiple camera positions. As described before, the detected specular line segments lie on the plane π_i joining the center of projection of the i^{th} camera and the axis of the screw as shown in Figure 3.7. The reconstruction of the 3D line corresponding to the axis of the screw is given by the intersection of these planes. In the remainder of this section, we present algorithms for reconstruction of the central axis for three view and two view configurations.

3.4.2.1 3D Line Reconstruction using Three Views

We first look into the well studied relation between a 3D line and its corresponding projections in multiple views. We then use a geometric constraint to find

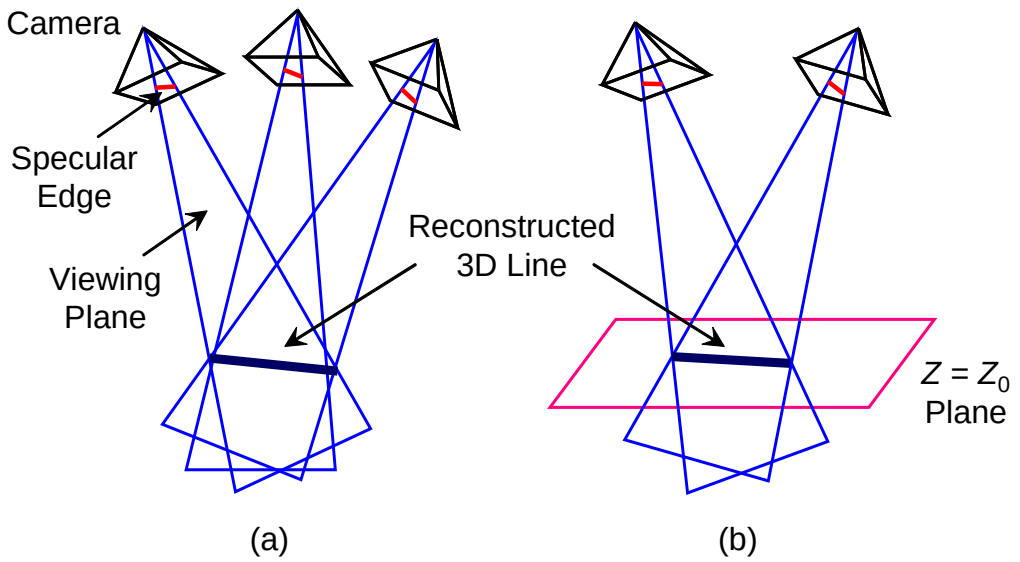


Figure 3.9: (a) Intersection of three planes becomes a line only if the correspondence is correct. (b) Two viewing planes and $Z = Z_0$ plane can be used for the two view case.

the lines corresponding to the same screw across three views and reconstruct the 3D line.

Geometry of Line Projection: Consider a line $\mathbf{L}$ in 3-space which is imaged in three views. Let l_i be the projection of $\mathbf{L}$ in view i and P_i be the projection matrix of view i. Using the projection matrix, an image point can be back-projected to a ray (passing through the camera center) in the world coordinate system. Both end-points of the line in the image plane can be back-projected as individual rays. Alternatively, the back projection of an image line l_i corresponds to a plane in view i as π_i

$$\pi_i \equiv A_i X + B_i Y + C_i Z + D_i = 0 \tag{3.4}$$

Finding Correspondences: Since there are multiple screws in the bin, we

need to identify the lines corresponding to the same screw from different views. In order to find correspondences, planes (formed by back projecting the lines) from all three views are transformed to a common coordinate system. This transformation is easily obtained as the camera poses are known. The common coordinate system is chosen as the camera coordinate system of the first view. We then use the property that three arbitrary planes do not meet at a single line to obtain our correspondences. This intersection constraint is expressed algebraically by the requirement that the 4×3 matrix defined by the coefficients of three planes [62]

$$\left(\underbrace{\begin{array}{ccc} A_i & B_i & C_i \\ A_j & B_j & C_j \\ A_k & B_k & C_k \end{array}}_{\mathbf{A}} \underbrace{\begin{array}{c} D_i \\ D_j \\ D_k \end{array}}_{\mathbf{b}} \right) \tag{3.5}$$

should have rank two. This constraint is satisfied if the determinants of four 3×3 sub-matrices of (3.5) are zero. In practice, due to noise in image measurements, even for lines in correspondence the determinants of these sub-matrices are not zero but small. We therefore compute as the correspondence cost the sum of four determinant values for each triplet of lines, and select the triplets that have the cost smaller than a threshold.

Reconstruction of the 3D Line: After finding the three lines corresponding to the same screw, we compute the 3D line passing through the axis of the screw. The line equation in 3D can be written as

$$\mathbf{X} = \mathbf{X}_0 + \lambda \mathbf{X}_1, \tag{3.6}$$

where $\mathbf{X}_0$ is a point on the line and $\mathbf{X}_1$ is the direction of the line. The direction

of the line should be computed as perpendicular to the all the plane normals as possible. This can be obtained by solving $\mathbf{A}\mathbf{X}_1 = 0$. The optimum solution (in least squares sense) is given by the right singular vector corresponding to the smallest singular value of matrix $\mathbf{A}$. We select $\mathbf{X}_0$ as the point which is closest to the three planes. This point can be found via the least squares solution of

$$\mathbf{A}\mathbf{X}_0 = -\mathbf{b}$$

.

Degenerate Configurations: If two planes are close to being parallel, the rank of the matrix (3.5) becomes close to two regardless of the other plane, leading to difficulty in finding correspondence. We therefore ignore such pairs of planes by checking the angle between the normals of the two planes. This primarily happens when the axis of screw is aligned with the translation between two camera positions. If we randomly choose three camera positions, all three translational directions between camera pairs become degenerate directions. To avoid this, we move the camera on a straight line so that there is only a single degenerate direction. In addition, we change this direction of motion to handle screws with varying poses during subsequent pickups.

3.4.2.2 3D Line Reconstruction using Two Views

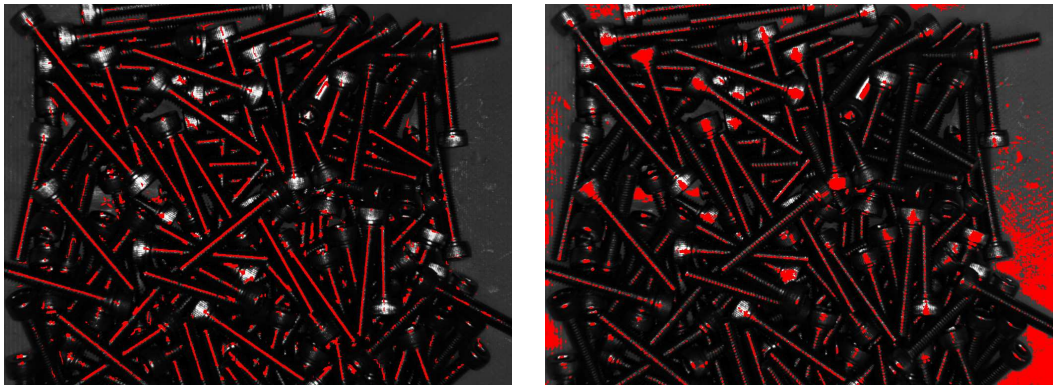
Since two non-degenerate planes from two views always intersect on a line, finding correspondences using two views is typically difficult. However, the correspondence can be computed if we assume that the 3D lines are located around some

plane in the world coordinate system. As shown in Figure 3.9 (b), we use $Z = Z_0$ plane together with the two viewing planes and compute the correspondence cost in the same way as the three view case. This Z_0 value is obtained by approximately measuring the physical distance between the camera and the bin with screws. The proposed cost favors screw (a) whose Z position is close to Z_0 and (b) whose angle is close to horizontal. After finding the correspondence, we reconstruct a 3D line as the intersection of two viewing planes (without using $Z = Z_0$ plane).

3.4.2.3 Position Estimation

After reconstructing the 3D line (3.6), we compute the segment of the line corresponding to the screw by back-projecting the end points of 2D lines in each view and finding the intersection point between the back-projected viewing ray and the reconstructed 3D line. We further validate the reconstructed line segment by comparing the length of the reconstructed line segment with the physical length of screw. The center of the line segment is chosen as the 3D gripping position.

Finally, we determine the end point of the line segment corresponding to the head of the screw. We compute the maximum image in all the views, which is simply a pixel-wise maximum over the 8 MFC images. The head of the screw is a smooth region with relatively lower curvature compared to the screw body. Therefore the specular highlights on the screw head move in a larger neighborhood with the alternating flashes and produce brighter patches in the maximum image compared to the screw tail.



(a) Proposed (b) Threshold

Figure 3.10: Comparison of the threshold-based highlight detection with the proposed algorithm. Detected edges superimposed (in red) on the image. Best viewed in color.

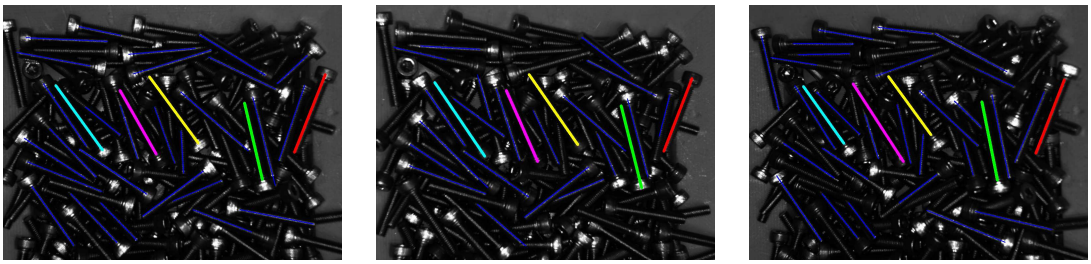
3.5 Experiments

We performed an extensive evaluation of the proposed method on the robot platform shown in Figure 3.1. We refer the readers to the accompanying video, which demonstrates bin picking of screws using the robot arm[1].

3.5.1 Specular Feature Detection

We first compare the proposed specular feature detection algorithm with the basic approach for detecting specularities in the scene using simple thresholding on the maximum image. The results are shown in Figure 3.10 where detected features are superimposed (in red) on the images. Threshold-based highlight detection is very sensitive and needs to be tuned every time scene or any parameter of scene

[1]The video was captured with unoptimized implementation. Please see Table 3.3 for the optimized system performance.



(a) View 1 (b) View 2 (c) View 3

Figure 3.11: Pixels in blue show the line fitting results on specular highlights detected from three views. Top 5 line correspondences across views have been shown respectively in green, yellow, cyan, magenta and orange.

e.g., illumination, viewpoint, etc. changes. Even with possibly the best choice of threshold, highlights from the glossy background cause a lot of spurious detections as seen in Figure 3.10(b). The proposed algorithm robustly detects specular features in the regions of high curvature as shown in Figure 3.10(a).

Figures 3.11(a)–3.11(c) show the correspondence results for three view approach. Pixels in blue show lines fit to the detected specular features. The top 5 line correspondences across three views are labeled with different colors. This visually validates the accuracy of our algorithm. The algorithm successfully locates multiple correspondences across views and hence can simultaneously estimate multiple poses within a single capture cycle. Figure 3.13 shows histograms of the correspondence costs for true and false matches. There is a large gap between true and false matches.

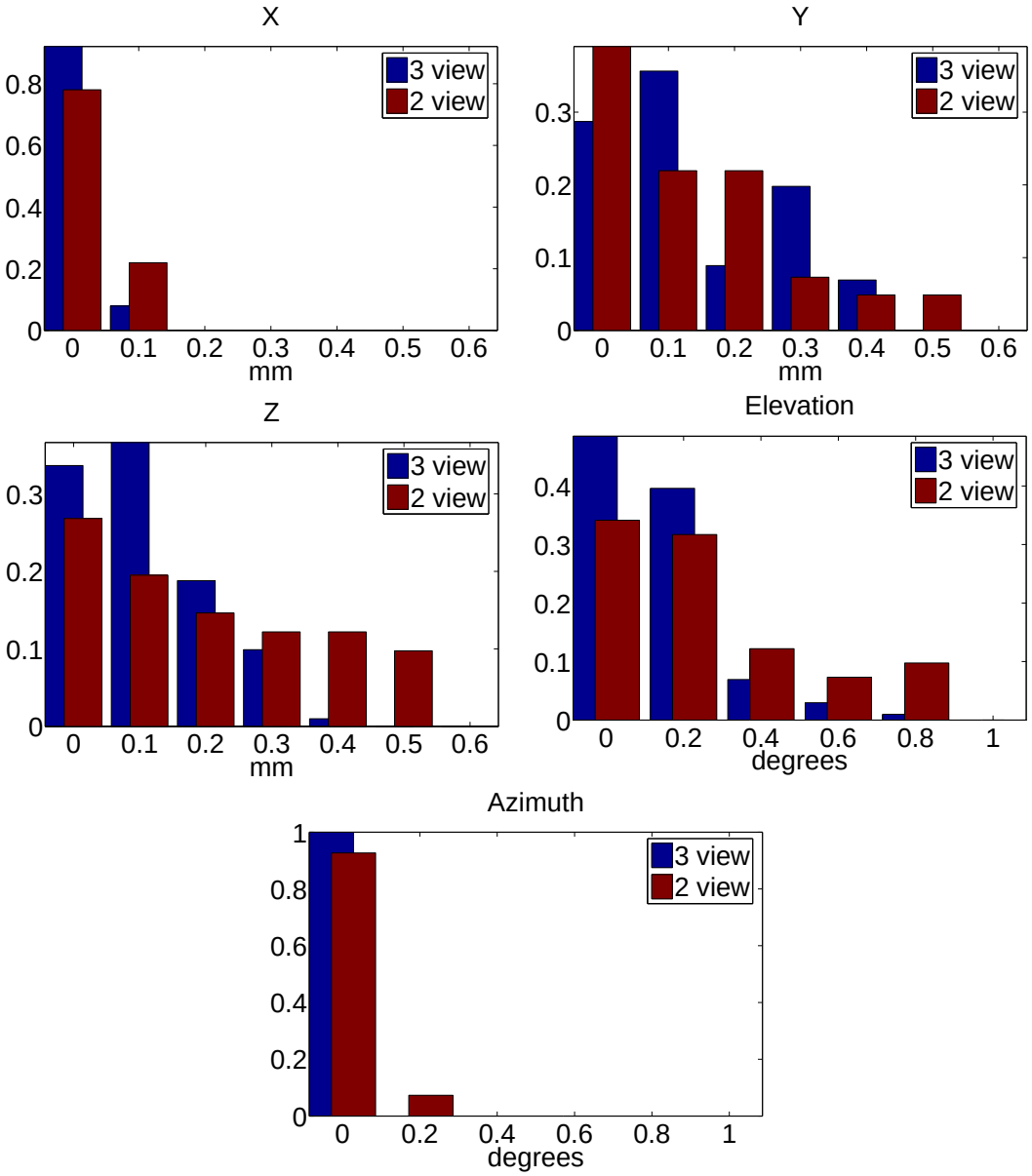
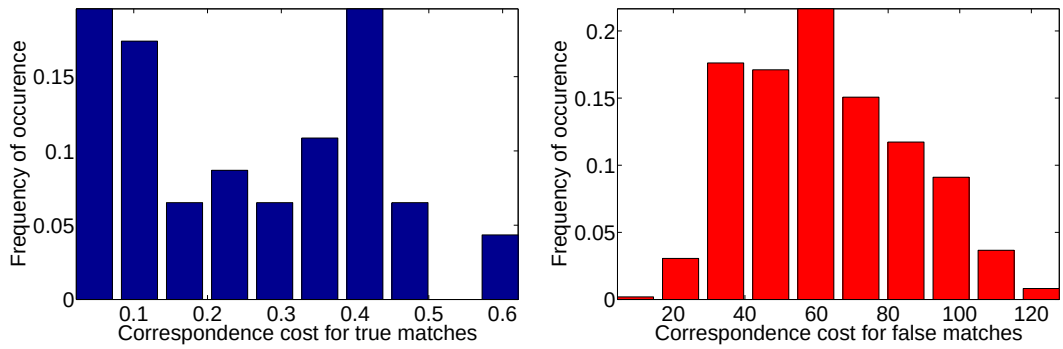


Figure 3.12: **Histograms of deviations of the pose estimates from their median.** Left three figures show errors in translations (X, Y, Z) and right two figures show errors in rotations. The 3 view approach (blue bars) has less deviation than the 2 view approach (red bars).



(a) Histogram for true matches

(b) Histogram for false matches

Figure 3.13: **The correspondence cost for true and false matches.** Note that the scale on the horizontal axis for false matches is 200× larger than that for true matches.

3.5.2 Single Screw Pose Estimation

In order to statistically evaluate the accuracy of the proposed system, we devised a method to evaluate the consistency of pose estimate irrespective of the viewpoint of the camera. Towards this evaluation, we first placed a single (isolated) M4 25 mm length screw in the bin. This is then imaged using MFC from 15 different locations arranged in a 3×5 grid. This setup produced 50 image pairs for 2-view and 100 image triplets for 3-view pose estimation where the camera positions have reasonable baseline with respect to the screw orientation. Since the object is static, the estimated pose of the object in the world coordinate system should be identical irrespective of the viewpoints. The median pose estimate of each method is selected as the ground truth.

Figure 3.12 shows the histogram of absolute deviations from the median pose estimates and Table 3.1 presents the mean absolute deviations. The results demon-

Average absolute error	X mm	Y mm	Z mm	Elevation degree	Azimuth degree
2-view	0.06	0.18	0.27	0.38	0.08
3-view	0.05	0.18	0.16	0.23	0.04

Table 3.1: Average absolute pose estimation errors for the 2-view and 3-view approaches.

	Trials	Pickup failure	Flip failure	Success rate
2-view	100	7	6	87%
3-view	100	4	5	91%

Table 3.2: Statistical comparison of screw pickup success rate between 2-view and 3-view in highly cluttered scenes.

strate that both (3-view and 2-view) algorithms result in highly consistent estimates with mean absolute deviation of less than 0.5 mm in all the three directions (X, Y, Z) and less than 0.8 degrees in the estimates of elevation and azimuth angles. In this experiment, the screw was placed perpendicular to X axis and the motion of the camera had larger baseline along Y axis, which resulted in slightly smaller translational error in X axis. The 3-view approach achieves smaller estimation deviations than the 2-view approach.

3.5.3 Bin of Screws

We then quantitatively evaluated the accuracy of the proposed system to detect and localize objects in highly cluttered scenes. More than 200 screws (M4, 25 mm length) were thrown together in a bin to create cluttered scenes just as shown in Figure 3.1(a). The gripper was made up of two vertical steel pins as shown in Figure 3.1(d). In its default position, the gripper is open with its pins apart. In order to pick up the screw, the gripper first moves to the estimated location/orientation and then closes the grip with the screw held horizontally between the two pins. This location is assigned such that the midpoint between the two pins coincide with the reconstructed axis and the gripper is perpendicular to the screw direction.

The width of the gripper, in its open position, is kept at 5 mm, which is 1 mm larger than the diameter of the screw. Therefore, to successfully grip the screw (before lifting the object) the error in pose estimate should be less than 0.5 mm. When the pose estimate error is greater than 0.5 mm the gripper fails to pick up the object. We achieved a 96% grasping rate for 3-view and 93% for 2-view approach evaluated over 100 trials each (Table 3.2). Among these successful grasps there were a few cases where the algorithm failed to correctly detect the head direction of the screw which resulted in pose flip failures. There were 5 flip failures in 3-view estimates and 6 flip failures in 2-view estimates, which resulted in overall correct pose estimates of 91% and 87% respectively.

The pickup failures were mainly caused when two screws (say S_1 and S_2) in parallel or anti-parallel orientations are very close to each other. In such scenario,

MFC Cap.	Spec. Edges	Line Fitting	Corres. (2view)	Corres. (3view)	Total (2view)	Total (3view)
0.87	0.11	0.15	0.01	0.04	2.3	3.4

Table 3.3: Average time required for each process of the proposed approach. The first 3 processes are repeated for 2 or 3 views. All numbers in seconds.

sometimes line correspondence ends up matching S_2 in one of the view to S_1 in the other two views. Hence, pose estimate is completely off and leads to the failure in grasping of the screws. The additional assumption about the $Z = Z_0$ plane in 2-view case increased the number of faulty correspondences and hence led to a slight deterioration in the success rate.

Complexity: Table 3.3 shows the average processing time required for each process of the proposed approach in extremely cluttered environments. This was measured on an Intel 3.06Ghz Quad-Core CPU with a C++ implementation. Capturing MFC images is the bottleneck of the current system. The processing times are only 0.53 and 0.82 seconds for the 2-view and 3-view approaches respectively, excluding the capturing time. Although in the current version the robot picks up a single screw for each capture cycle, the algorithm simultaneously estimates poses of multiple screws. It is possible to pick up multiple screws within a single capture cycle, which would drastically accelerate the system.

3.6 Conclusion and Future Work

We presented a system for specular object detection and pose estimation using a multi-flash camera. We introduced a novel feature that is present on the high curvature regions of specular objects. We developed an algorithm to reliably detect these features by exploiting the changing lighting positions. We demonstrated our approach for pose estimation of screws in a highly cluttered bin, where we used specular features detected at multiple views to reconstruct the 3D axis of the screw. We implemented our algorithm on a robot arm and achieved highly accurate pose estimation with location and orientation errors less than 0.5 mm and 0.8° respectively. The proposed specular feature is useful in detecting any shiny object with high curvature. Developing pose estimation algorithms using specular features for a wider class of objects would be an interesting future direction.

Chapter 4

Video Précis: Highlighting Diverse Aspects of Videos

4.1 Introduction

Recent years have witnessed a tremendous increase in multimedia content driven by inexpensive video cameras and the growth of the Internet. The amount of visual data being recorded and accessed has been increasing with the rising popularity of several social networking and video sharing websites. In several applications, there is a need to gain a quick overview of the contents of a video. As an example, imagine having to browse through an hour of skating video to view all the diverse clips corresponding to camel spin, axel-jump, etc. by simple linear browsing. Instead, if one could extract/filter a few segments of the original video, such that they are "mutually exclusive and exhaustive", this will result in significant improvement in user experience. Similarly, in video-based security and surveillance systems, analysts spend long hours sifting through large video recordings of parking lots and airports to locate events of interest. The ability to gain a quick overview of diverse aspects of hour-long videos is vital in these applications.

Recently, with the advent of improved multimedia technologies (inexpensive cameras, video hosting websites), unconstrained videos have become commonplace. Consider a few examples of videos shown in Figure 4.1. Each of these videos has

Figure 4.1: Examples of some unconstrained videos. Each row here shows 6 frames from videos representing various domains: (a) Cheerleading YouTube video, (b) "Office Tour" YouTube video, and (c) aerial video.

more than 5000 frames (each video is of duration 4 minutes). As can be seen, there exists a lot of diverse information in each of these videos. This calls attention to the problem of optimally selecting 'exemplars' from the video so as to obtain a quick overview of the video without losing any detail. Here, we define exemplars as "informative/key segments" of the video selected to provide condensed and succinct representations of the content of a video. An exemplar could be a single frame or a video segment (continuous sequence of frames) of fixed length depending upon the end application.

This ability to produce an abstract of the video, without losing the details, requires addressing two conflicting requirements – 1) the summary should be representative of the video ("exhaustive"), and 2) the summary should highlight diverse aspects of the video ("mutually exclusive"). The first criterion suggests that the summary should 'cover' most of the video in terms of global representation. The

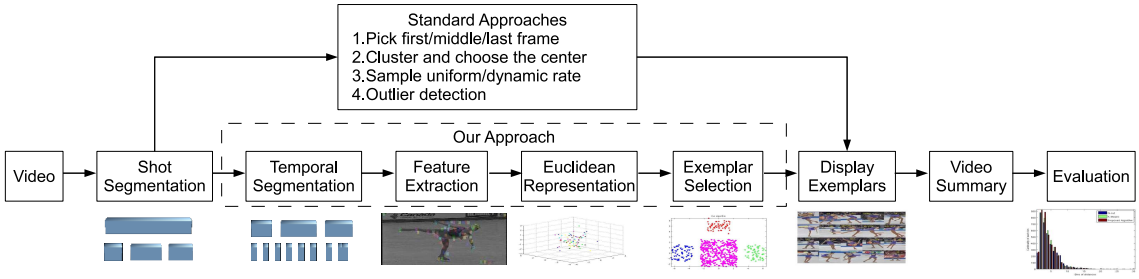


Figure 4.2: Overview of existing approaches to video summarization. Existing approaches rely predominantly on computing and processing shots and end up picking exemplars from the dominant event leaving the interesting but infrequent events.

second criterion suggests that the elements of the summary should be as distinct as possible. In this chapter, we propose a novel formulation to quantify this trade-off and subsequently propose an approach for generating a summary that takes into account these two important considerations. We provide a mathematical formulation of these criteria using a unified cost function, and propose an algorithm to solve it.

4.2 Related Work

The problem of generating the summary given a video has attracted significant attention especially over the past few years. Several video abstraction systems have been proposed and good recent surveys with systematic classification of various approaches can be found in [63, 64, 65, 66, 67, 68]. An overview of the existing approaches is given in Figure 4.2. Most existing approaches have relied predominantly on computation and processing of 'shots'. A shot is a sequence of frames within a continuous capture period and the transition between two consecutive shots is termed as the shot boundary. These shot boundaries are then used to detect the

temporal span of each shot. Thereafter, exemplars within shots are selected using simple techniques such as the first/middle/last frames or merely random sampling. These approaches were motivated from and tuned to genres such as movies and news-videos where the use of shots is an integral part of the video capturing process. In this better understood genres of newscasts and movies, the presence of shots provides cues to 'interesting' content. Thus, simple techniques such as preserving the shot-boundaries worked reasonably well in such genres to provide an overview. Examples of this approach include [69, 70] who use shot boundary detectors and then use the first/middle/last frames from each shot as the summary. Similarly, Zhang *et al.* [71] choose the first frame of each shot as a keyframe, and then if the difference between the subsequent frames and the latest keyframe exceeds a threshold, the current frame is also chosen as a keyframe.

However, in unconstrained videos such as the ones shown in Figure 4.1, within shot variations are too large to be ignored. For instance, each of the videos shown in Figure 4.1 come from a *single shot* video containing 5000 frames. Therefore, a single frame per shot would not be able to capture these variations. So, multiple keyframes/segments must be assigned to each shot-based on the dynamics of the shot. Zhang *et al.* [72] and Gunsel *et al.* [73] segment the video into shots and select the first clean frame of each shot as a keyframe. Other frames in the shot that are sufficiently different from the last keyframe are marked as keyframes as well. This shot-based approach captures the original video well when the shots are largely stationary or unchanging. Another approach that was used to solve this problem was to use some form of smart-sampling method that could pick more frames from highly

informative regions. Nam and Tewfik [74] generate the summary by adaptive and dynamic sampling of the video sequence. Specifically, they calculate sub-shot units of the video sequence as well as motion activity for each of them. Sub-shot units with high quantized motion intensity index are then sampled at a higher rate than those with a lower index. Divakaran *et al.* [75] also propose motion-based nonuniform sampling of the frames. The underlying hypothesis of motion-based sampling is that the intensity of motion activity of a video segment is a direct indication of its "summarizability". In [76] and [77], the authors use audio descriptors along with the motion descriptors to summarize the videos. Recently, Chen *et al.* [78] constructed the relational graph using the correspondence between video and text information, and then exploited the graph entropy model to detect meaningful shots and relations. De Menthon *et al.* [79] represent the video as a curve in a high dimensional feature space and use a multidimensional curve splitting algorithm to linearize the curve and extract the keyframes. The limitation with this class of approaches is that these measures of summarizability are local in nature. In effect, this may lead to important segments being missed while longer segments might show multiple frames with similar content.

To remedy these issues, one needs a principled way to select informative exemplars from all the available frames or video segments. The simplest of approaches would be to use clustering methods. In this approach, video frames are treated as points in a feature space and then the most representative frames/segments are shown from each cluster as part of the video abstract. Various image-based features like color, motion, etc. have been employed for this task. Subsequent to clustering,

cluster centers are presented to users as the video summary. This also enables a user to search for relevant clips in the original video by using the clusters as an index. Ferman *et al.* [80] and Zhuang *et al.* [81] use clustering on the frames within each shot. The frame closest to the center of the largest cluster is selected as the keyframe for that shot. This approach does not capture within shot variations of non-stationary shots well, as only one keyframe was chosen per shot. Hanjalic and Zhang [82] propose another clustering technique where all video frames are clustered into a variable number of clusters. Cluster validity analysis is then performed to determine the optimal number of clusters n. Similarly, activity specific features are extracted over small segments of videos, and each segment is represented as a collection of features derived from a histogram as in [83] or as a linear dynamical system [84]. The methods presented in [85][86] have also focused on generating summaries for domain-specific videos where special features using the domain knowledge can be employed. Given a long video, these clustering approaches proceed in an unsupervised fashion to extract summaries.

Although, clustering techniques have been quite effective, many of them reduce to choosing representative frames from the most dominant clusters. However, as discussed in [87], in a large class of videos (e.g., surveillance, sports, etc.) interesting events happen infrequently among the background of usual events. Many times these unusual/interesting events are the desired events which are often not captured by traditional clustering approaches. To resolve this issue, another approach was to choose outliers (unusual or uninteresting events) as the exemplars as [87]. However, unusual events like a commercial break in a soccer video need not be interesting

to the end user [87]. This motivates us to propose a summarization method that ensures mutually exclusive exemplars which are also exhaustive.

To complete the discussion, we briefly discuss visualization methods for video summaries. Significant research has also been done on finding novel ways of presenting and visualizing the summary. One of the most commonly used methods to present summaries is via storyboards. Another approach is through mosaicing [88][89][90], which tries to present most of the pixel intensity variations that are spread out over several minutes by piecing together a large mosaic. Another class of approaches collapses the regions of motion into a smaller spatio-temporal volume. In [91], this was done by detecting 'tubes' of moving objects in the video and fitting all the tubes into a coherent video. This approach essentially displays all the moving objects in the scene with no special regard to what activity is being performed by the object. A content-aware resizing approach is taken in [92], where 'seams' of low-gradient are successively removed from a video. The resulting video then represents high-gradient information in the video. Another approach for re-targeting the image/video data has been presented in [93]. All these methods are primarily visualization techniques and are best suited for creating visual 'thumbnails' of videos. In this work, we restrict ourselves to the problem of choosing 'good' exemplars to represent the unconstrained video (of any domain) well and generate a quick overview of the video. The focus here is not on how the summary is visualized as we simply choose to use one or the other method as appropriate in experiments.

4.3 Contributions

In this chapter, we make the following specific contributions.

- We propose two criteria 'coverage' (exhaustive) and 'diversity' (mutually exclusive) that video summarization should embody.
- Mathematical formulation of these criteria leading to a cost function for video summarization. An algorithm is proposed to optimize this criteria.
- Further, to demonstrate the effectiveness of the proposed criteria, results on four different classes of videos are presented.
- Subsequently, the summary generated is evaluated using both quantitative measures and user-studies.

Organization of the chapter: In section 4.4, we first discuss the proposed video summarization framework. Section 4.5 discusses the role of the proposed criteria in video summarization. Then, in section 4.6, we quantify these criteria and provide a mathematical formulation for the summarization problem, followed by a discussion of how to solve the summarization problem in section 4.7. In section 4.8, we propose a few extensions to the proposed cost function. Section 4.9 presents the algorithm for preprocessing and feature extraction from the video. In section 4.10, we provide several experimental results that demonstrate the effectiveness of the proposed approach. In sections 4.11 and 4.12 we evaluate our method using quantitative and qualitative measures respectively. Finally, benefits and limitations of the proposed approach are provided in section 4.13.

4.4 Video Summarization Framework:

Before proceeding further, we first discuss the proposed framework for summarizing video content, from which specific algorithms can be derived. Video summarization can be seen as proceeding in a sequence of steps, which are discussed below and concisely shown in Figure 4.2. We also discuss here various considerations at each step.

- **Temporal Segmentation:** A given video first needs to be broken into segments where there is some consistency in viewpoint, scene, etc. This frequently takes the form of shot segmentation. Each shot is further segmented into small video segments. This segmentation should ensure that each of the segments is smooth with minimum variation and consistency of dynamics/view/objects in it. In our implementation, we choose these segments to be of constant length of 20 frames. Choosing such short segments ensures the consistency in each segment.

- **Feature Extraction:** Once coherent shots are identified, we need to extract concise features that capture relevant information from the frames. The video segments are then represented as points in an N-dimensional space. Towards this goal, features are independently extracted from each segment. If the end-application requires the summary to capture the content in terms of various spatial appearances and dynamics, features such as the spatio-temporal features suggested in of [3] may be chosen. Further, if the property of rate/view/scale invariance is to be incorporated in the summarization system,

features with such properties should be chosen. If the summarization system has to use audio cues, features from audio should be extracted [76][77].

- **Exemplar Selection:** This stage selects important exemplars in the video that capture the essence of the video. Given a set of points in an Euclidean space, the problem of summarization boils down to one of optimally selecting a subset of points (exemplars). At this stage, the measure of optimality becomes crucial. Traditionally, this has been implemented by means of standard clustering approaches like k-means, N-cut, etc. Alternately, optimizing a specified cost function that incorporates desired attributes over the chosen subset [94] can be used for this task. In this work, we adopt the latter approach. We first discuss two criteria 'coverage' and 'diversity' that a summary should have and then incorporate them into a cost function. Traditionally, as discussed in the related work section, clustering approaches have been adopted for choosing this optimal subset. Hence, to show the significance of the two criteria, we compare it with the two well known clustering approaches – k-means and N-cut.

- **Visualization:** The best visualization of this optimal subset (i.e., displaying the exemplars) depends largely on the end-application. It can be displayed as a static storyboard or as a temporal concatenation of the chosen video segments. In the latter case additional criteria like 'coherence' and 'comprehensibility' can be incorporated to make the visualization smoother. However, the focus of this work is to optimally select the exemplars and we choose here the

simplistic way of temporally concatenating the chosen exemplars.

4.5 Role of Diversity and Coverage in a Précis

Précis is a term that is used to refer to summaries of long documents, that encompass the essence of a document without omitting any significant details. This requires the summary to contain the dual properties of diversity and coverage. While easy to state, it is not apparent what this entails in computational terms. Before delving into mathematical details, let us consider what these ideas mean in terms of video summarization. In a large class of videos (e.g. surveillance videos, sports videos, etc.), large portions typically contain 'uninteresting' events. Thus, standard summarization approaches might miss the really interesting aspects and preserve the dominant uninteresting portions. Most of the clustering- based approaches do not necessarily enforce the criterion that the discovered summary highlight diverse aspects of the video. Further, even if by some pre-processing steps one were to remove the irrelevant parts of a video, the results of optimal subset selection can be easily skewed by uneven statistical distribution of the video segments. An action/event that occurs more frequently than others would skew the chosen subset towards this more dominant event. Thus motivated by these consideration, we suggest two properties for a good summary of videos a) coverage and b) diversity. The first criterion suggests that the summary should 'cover' most of the video in terms of global representation. The second criterion suggests that the elements of the summary should be as distinct as possible.

In this chapter, we explore these ideas for the problem of video summarization. To solve this problem, we model the summarization process as searching for a set of video segments from the original video that satisfy the above discussed properties. We provide a conceptually simple quantification of these criteria into a cost function. We show how to find a good summary by a combinatorial optimization approach to minimize the cost function. Experiments show that even with simple and intuitive metrics as proposed here, significant improvements in summarization quality over traditional approaches are obtained. Further, the problem formulation and solution is largely independent of the choice of features and easily extends to new features and thereby to new classes of videos.

It is worthwhile to study the parallel advances in the document mining community which has also addressed similar problems. In fact, the widely used bags-of-words model and topic models have their origins in text-mining and retrieval. One of the important problems in this field is to generate a summary of a single long document by selecting informative sentences [95]. By enforcing that the document summary should contain the properties of 'Coverage' and 'Orthogonality' [96], the generated summaries were found to be of significantly higher quality. However, the methods proposed in [96] are strongly tied to documents and do not easily generalize to other domains such as videos.

4.6 Coverage and Diversity: A Cost Function

Consider a video V as a collection of video segments $V = \{v_1, v_2, \ldots v_n\}$. Let us assume that there exists a representation of each video segment that maps it into a Euclidean representation $\mathbb{F} = \{F_1, F_2 \ldots F_n\}$ where each $F_i \in \mathbb{R}^d$ for some d. This representation could be one of several choices such as bags-of-words [97], 3D structure tensors [83], motion history [98], etc. For manifold-valued representations such as linear dynamic models [99] or covariance matrices [100], we shall assume that we can obtain an Euclidean representation via the logarithmic map on the manifold. Thus, the mapping from video segments to the Euclidean representation can be made flexible and we do not tie down the method to any specific choice of features.

Once we have made the choice as to what representation is best suited for a class of videos, we can then proceed to quantify and optimize the criteria for summarization. We define a summary S of length K, to be a collection of K segment indices $S = \{\alpha_1, \ldots \alpha_K\}$, where $\alpha_i \in \{1, \ldots n\}$ for all i. Now, we consider the set $S_F = \{F_{\alpha_i} \,|\alpha_i \in S\}$ as the set of summary centroids. The centroids implicitly partition the space of Fs by assigning each F_i to the closest centroid. Let V_{α_i} be the partition corresponding to F_{α_i}, i.e. V_{α_i} are the elements of $\mathbb{F}$ which are closer to F_{α_i} than any other centroid. The *scatter* of V_{α_i} can then be defined as

$$scatter(V_{\alpha_i}) = \sum_{F_k \in V_{\alpha_i}} (F_k - F_{\alpha_i})(F_k - F_{\alpha_i})^T \tag{4.1}$$

Then, the *coverage* of the summary S can be measured in terms of squared error

$$err(S) = tr\left[\sum_i scatter(V_{\alpha_i})\right] \tag{4.2}$$

$$= tr\left[\sum_i \sum_{F_k \in V_{\alpha_i}} (F_k - F_{\alpha_i})(F_k - F_{\alpha_i})^T\right] \tag{4.3}$$

A high coverage implies a low error, and vice versa. On the other hand, the *diversity* of the summary is measured in terms of the scatter of the centroids as follows.

$$div(S) = tr\left[\sum_i (F_{\alpha_i} - \overline{F})(F_{\alpha_i} - \overline{F})^T\right] \tag{4.4}$$

where $\overline{F} = \frac{1}{K}\sum_j F_{\alpha_j}$, is the mean of the centroids. From this measure of diversity, we can define the 'redundancy' of the summary as $red(S) = D - div(S)$, where D is an upper bound on the diversity possible. This measure ensures that $red(S)$ is always a positive number. However, the precise value of D is not critical for the rest of the discussion. Using, these measures of coverage and redundancy, the overall cost of a summary is given by

$$J(S) = \alpha * err(S) + (1 - \alpha) * red(S) \tag{4.5}$$

with $0 \leq \alpha \leq 1$. The goal of the summarization algorithm is then to minimize the cost-function $\hat{S} = min_S J(S)$. Here α represents a weighting parameter that provides relative weightage to each of the criterion. A value of $\alpha = 1$ boils the problem down to a standard least squares problem which can be solved by k-means or k-medoids. We let the value of α to be a user defined parameter whose value should be decreased if the diversity criterion is to be emphasized. Note that the precise value of D does not affect the minimization of the cost-function, since it

only adds a constant value to the cost. It provides a convenient method to pose the problem as the minimization of a positive function.

4.6.0.1 Relation with other cost functions

Here we discuss the relationship between the cost function in (4.5) with other well known cost functions. k-means and k-medoids minimize a cost function which is similar to the first part of (4.5). The difference is the diversity criterion, in the absence of which this subset selection can be easily skewed toward dominant events. The diversity criterion ensures that even infrequent but interesting events are represented in the summary. Affinity propagation [101] is a relatively new algorithm which also minimizes the sum of squares error criterion, but does not require initialization. Normalized-cuts (Ncuts) [102] is a graph-partitioning algorithm which searches for sub-graphs with large 'internal association' normalized by its association with the rest of the graph. The internal association can be interpreted as being similar to the coverage criterion. But the normalization term in Ncuts forces the algorithm to find balanced clusters. This does not ensure that the cluster centroids highlight diverse aspects of the video.

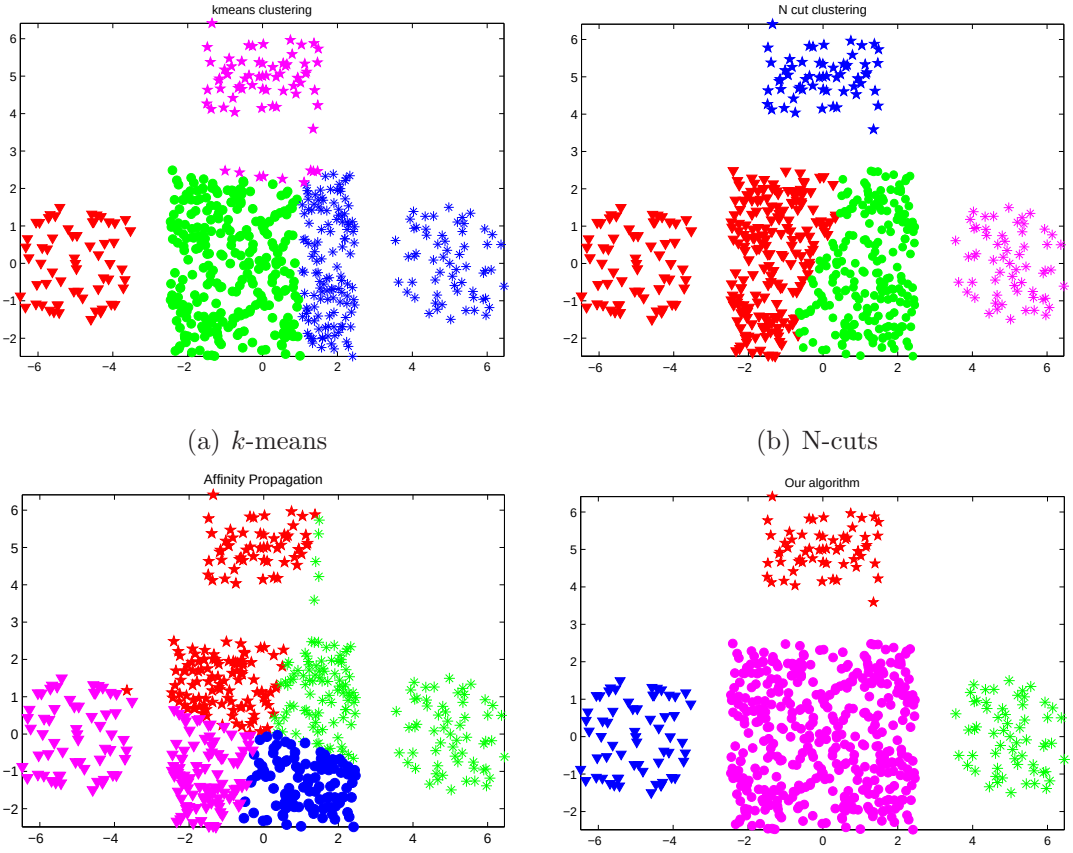
The diversity criterion looks similar in form to the 'between-class' scatter criterion and coverage is similar in form to the 'within-class' scatter used in Fisher's Linear Discriminant Analysis (LDA), but differs from it in several significant ways. Firstly, there are no 'class labels' in the current scenario. Secondly, we are not interested in searching for projections of the data as is the case in LDA. Further,

if maximizing diversity ($\alpha = 0$) were the only criterion, then the proposed method would be sensitive to outliers. However, the coverage criteria prevents the proposed cost function from latching on to outliers in the data.

We illustrate these ideas using a synthetic experiment for optimal subset selection from a set of points in $\mathbb{R}^2$. For this experiment, we synthetically generated points in $\mathbb{R}^2$ in 4 distant sets. One of the sets was deliberately made larger than others. Then, we used standard approaches k-means, Ncuts, and Affinity Propagation to choose four exemplars. We show the results in Figures 4.3(a) - 4.3(c). As can be seen, due to the dominance of the central set, standard cost functions pick multiple exemplars (centroid) from the same set. On the other hand, the results of exemplar selection using the proposed cost-function are shown in Figure 4.3(d). It can be seen that the proposed cost-function finds a diverse set of exemplars.

4.7 Optimizing the Cost Function

In this section, we discuss how to optimize the cost-function $J(S)$ described in section 4.6. First, we observe that this is a combinatorial optimization problem where we need to find optimal solutions (not necessarily unique) in the (discrete) problem space. To find an optimal solution in a combinatorial optimization problem, the difficulty arises from the fact that there is a lattice of feasible points. In some of the other optimization problems like linear programming, this is not the case as the feasible region is a convex set. In fact, if we replace the least squares cost function of the classical k-medoids partitioning problem with the proposed cost-function, then



(a) k-means

(b) N-cuts

(c) Affinity Propagation

(d) Proposed Method

Figure 4.3: Comparison of the standard approaches and the proposed cost function in choosing exemplars. Each set is represented in a different color and the centroid of the set is chosen as the exemplar. Standard approaches do not find 'diverse' exemplars.

we note that both problems are similar in complexity. As the k-medoids problem is known to be NP-hard. Hence, the current problem is also NP-hard. This motivates the need to solve this problem by an approximate method. Observing the similarity between the current problem and classical k-medoids partitioning problem, we adopt the iterative procedure discussed in Algorithm 4.1.

We note that at each iteration the algorithm chooses a solution that either

Algorithm 4.1: Algorithm for minimizing the proposed cost function

Start with a random initialization of k centroids $S_F = \{S_{F_1}, S_{F_2}, .., S_{F_k}\}$.

Assign each of the remaining points in $\mathbb{F}$ to the nearest point in S_F.

Randomly select a point $S_{F_{rand}} \in \mathbb{F} \setminus S_F$.

Compute the difference $J_{swap} = J_{new} - J_{old}$ that results by swapping $S_{F_{rand}}$ with S_{F_j} where j is a randomly chosen index.

If $J_{swap} < 0$, replace S_{F_j} with $S_{F_{rand}}$. Else retain the original S_F.

Repeat steps $1.2 - 1.5$ till convergence or if maximum iterations are exceeded.

decreases the cost or leaves it unchanged. Hence, the optimization always proceeds in a 'greedy' fashion. Thus, it is likely that the solution may end up in a local minima. The next question to address is how many iterations are needed before the algorithm converges. We do not have a clear answer for this question, but it was observed that as the number of points increases the iterations required to converge increase exponentially. Further, due to the 'greedy' nature, the final solution depends on the initialization. Thus, to address these issues we experimented with several random initializations. For each initialization, we let the algorithm run for a large number of iterations ($\sim 10^3$). From the set of solutions thus obtained, we choose the one with the minimum cost. We found that this strategy works well in most cases.

Computational Complexity: If n is the number of data-points, k is the number of exemplars to be selected and t is the number of iterations, then the computational complexity of algorithm is O(k(n-k)t). Since, $n >> k$, the complexity is linear all three variables.

4.8 Weighted Kernel Coverage and Diversity

In many cases, features are usually not linearly separable in the feature space, but may require non-linear separating boundaries. In the existing literature, this problem is commonly solved in a reproducing kernel Hilbert space (RKHS) instead of in the original feature space such as in [103]. The basic assumption is that there exists a mapping into a high dimensional RKHS, where linear separability is possible. The problem is implicitly solved in RKHS via a suitably defined Mercer kernel. In this section, we shall discuss a kernel extension of the proposed cost function that enables us to optimize diversity and coverage in the RKHS.

Let $\phi_i = \phi(F_i)$ be the mapping of $F_i \in \mathbb{R}^d$ to the higher dimensional RKHS space $\mathcal{H}$. Let P_{α_i} be the partition of the points in the $\mathcal{H}$ space.

$$err(S) = \left[\sum_i \sum_{\phi_k \in P_{\alpha_i}} w_k ||\phi_k - \phi_{c_i}||^2 \right] \tag{4.6}$$

where, w_k are non-negative weights and c_i represents the index of the centroid of the i^{th} partition (P_{α_i}) in the $\mathcal{H}$ space.

$$\begin{aligned} c_i &= arg\ min_{c_i} \left[\sum_{\phi_k \in P_{\alpha_i}} w_k ||\phi_k - \phi_{c_i}||^2 \right] \qquad s.t. \quad \phi_{c_i} \in P_{\alpha_i} \\ &= arg\ min_{c_i} \left[\sum_{\phi_k \in P_{\alpha_i}} \left(w_k \phi_k^T \phi_k \ + w_k \phi_{c_i}^T \phi_{c_i} \ - 2 w_k \phi_k^T \phi_{c_i} \right) \right] \\ &= arg\ min_{c_i} \left[\sum_{\phi_k \in P_{\alpha_i}} \left(w_k K_{kk} \ + w_k K_{c_i c_i} \ - 2 w_k K_{k c_i} \right) \right] \end{aligned}$$

where, K is the kernel Gram matrix with $K_{ij} = \phi_i^T \phi_j$.

$$err(S) = \left[\sum_i \sum_{\phi_k \in P_{\alpha_i}} \left(w_k \phi_k^T \phi_k \ + w_k \phi_{c_i}^T \phi_{c_i} \ - 2 w_k \phi_k^T \phi_{c_i} \right) \right]$$
$$= \left[\sum_i \sum_{\phi_k \in P_{\alpha_i}} \left(w_k K_{kk} \ + w_k K_{c_i c_i} \ - 2 w_k K_{k c_i} \right) \right]$$

The *diversity* of the summary in the $\mathcal{H}$ space is measured similarly in terms of the scatter of the centroids as follows.

$$div(S) = \left[\sum_i w_{c_i} ||\phi(c_i) - \overline{p_c}||^2 \right] \qquad where, \quad \overline{p_c} = \frac{\sum_i w_{c_i} \phi_{c_i}}{\sum_i w_{c_i}}$$
$$= \left[\sum_i w_{c_i} (K_{c_i c_i} - \frac{2 \sum_j w_{c_j} K_{c_i c_j}}{\sum_j w_{c_j}} + \frac{\sum_{c_j, c_l} K_{c_j c_l}}{(\sum_j w_{c_j})^2}) \right]$$

Then, defining the redundancy of the summary as earlier $red(S) = D - div(S)$, the overall cost function $J(S)$ is given by

$$J(S) = \alpha * err(S) + (1 - \alpha) * red(S) \tag{4.7}$$

As can be seen, the mapping ϕ need not be computed explicitly, as the resulting cost-function and its optimization only require the knowledge of dot-products in the RKHS. These dot-products can be evaluated using a Mercer kernel. This kernel extension can be used when no separating hyperplane exists in the feature space and a "better" linear partition can be found in a high dimensional space. The weights can be uniform if no prior knowledge/preferences exist about the summary of the given video. In a relevance feedback scenario, the user can choose more relevant exemplars from the initial set of returned exemplars, which can be used to refine the summary using the above formulation by assigning higher weights to the chosen exemplars. The experiments presented in this chapter simply use $K_{ij} = x_i^T x_j$. This

worked well in all the experiments. However, the results derived in this section provide a principled means to optimize the performance to even higher levels. But we do not explore this avenue further in the experiments.

4.9 Video Analysis and Feature Extraction

In this section, we discuss the pre-processing of the video content and how we obtain the Euclidean representation of each video segment. Before we discuss the specific choices made, we would like to emphasize that the optimization cost-function and algorithm for exemplar selection are independent of the choices made in lower-level modules.

4.9.1 Feature Extraction and Euclidean Representation

In the first step, we perform video segmentation or shot segmentation, i.e., partition the video sequence into shots. Once the shots are detected, we extract spatio-temporal features for the Euclidean representation of the video. For this, we adopt the widely used bag-of-words approach to represent the video segments. Each video segment is assumed to be 20 frames long (though a multi-scale version is possible here). In the bag-of-words formalism, a long video is considered to be a 'document' or as a collection of sentences, where each segment corresponds to a 'sentence'. In turn, each sentence is considered as a vector in an N-dimensional Euclidean space with the i^{th} value as the indicator of the i^{th} word of the codebook. The words are learnt by clustering descriptors extracted around interest points from

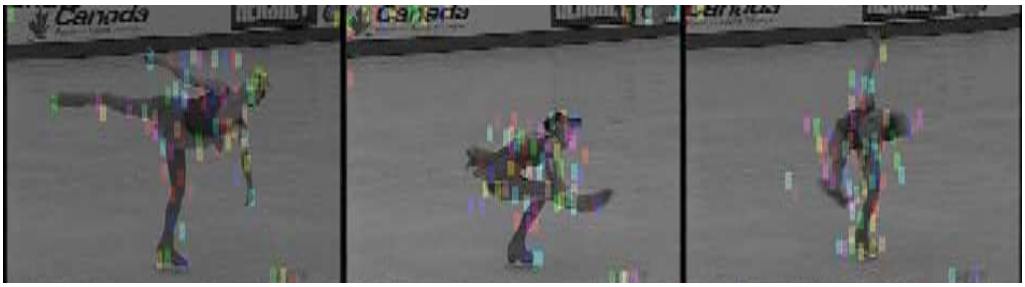


Figure 4.4: Example frames from figure skating [2] video sequence. Spatio-temporal patches (extracted using [3]) shown overlaid on the images.

the entire long video. To extract the interest-points, we used the method proposed in [3]. These interest points are the local maxima of the response function

$$R = (I * g * h_{ev})^2 + (I * g * h_{od})^2$$

extracted from a sequence of images. Here, I is the sequence of images, $g(x, y, \sigma)$ is the 2D Gaussian smoothing kernel applied along the spatial dimensions and h_{ev} and h_{od} are a quadrature pair of 1D Gabor filters applied temporally. The extracted cuboids are further processed by simply flattening them as suggested in [3]. The detector parameters are set to $\sigma = 2$ and $\tau = 2.5$. As described in [97], we build our codebook by clustering these flattened cuboids into $N = 500$ clusters. The bag of words approach is scalable to long video sequences.

Another question that remains to be answered is how do we decide the number of exemplars that should be chosen per shot. Any appropriate measure on the length of the summary would be on the whole video rather than each shot. Intuitively, a shot with higher variation should be represented with more exemplars than a shot with lesser variation. So, the number of key video segments chosen per shot is

decided using the covariance matrix of the shot. The motivation behind using the covariance matrix is because it captures the 'variation' within a shot well, hence can be used to assign the number of key video segments. The covariance matrix is calculated for each shot, denoted by $C(shot_i)$, using the Euclidean representation of each video segment in the shot. This matrix is then used to calculate the number of exemplars per shot:

$$K_{shot}(i) = \lceil \frac{tr(C(shot_i))}{\sum_i tr(C(shot_i))} * K_{video} \rceil$$

where $tr(C(shot_i))$ represents the trace of the covariance matrix of the shot. $K_{shot}(i)$ is the number of exemplars for the i^{th} shot while K_{video} is the number of exemplars for the video. $\lceil . \rceil$ represents the standard ceiling function.

4.10 Experiments

In this section, we describe experiments demonstrating video summarization using the proposed algorithm and a few illustrative comparisons. We provide experimental results on four representative videos: a) Constrained KTH human action dataset [4], b) Unconstrained figure skating dataset [2], c) Unconstrained aerial surveillance VIVID dataset, and d) An office-tour video from Youtube. [1]

4.10.1 Constrained KTH Action database

The first experiment was performed on the publicly available KTH human action dataset [4]. The dataset consists of six human actions (walking, jogging,

[1] Video results are also available at url **http://www.umiacs.umd.edu/users/nshroff/Precis.html**

Figure 4.5: Example frames from video sequences in the KTH dataset [4]. Figure reproduced from [4].

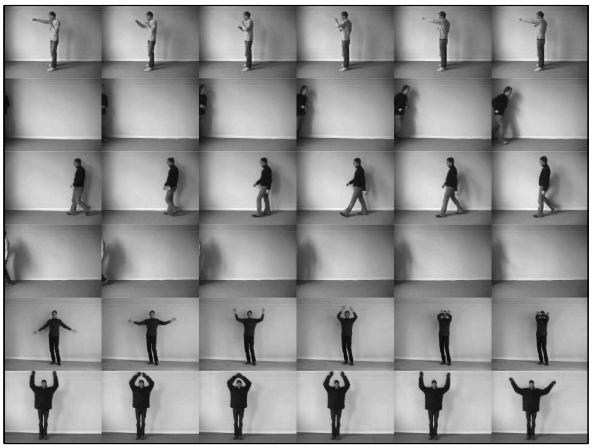
running, boxing, hand waving, and hand clapping) performed several times by 25 subjects in four different scenarios (backgrounds) as shown in Figure 4.5. In our experiment, we constructed a long video by appending a subset of this dataset which includes the first 5 subjects performing each of the six actions in the scenario *s4*.

The set of vectors was constructed as discussed in the section 4.9.1 which is then fed to the optimization algorithm discussed in section 4.7. The value of $\alpha = 0.45$ was used. Since there are precisely 6 distinct activities in the whole video, we generated a summary of length $K = 6$ to test the strength of the proposed framework. Ideally, each element in the generated summary should represent a distinct activity. The results of the summarization are shown in Figure 4.6(c). Each row represents a video segment of the generated summary. Due to space constraints, we only show 6 uniformly sampled frames from the 20 frames of each

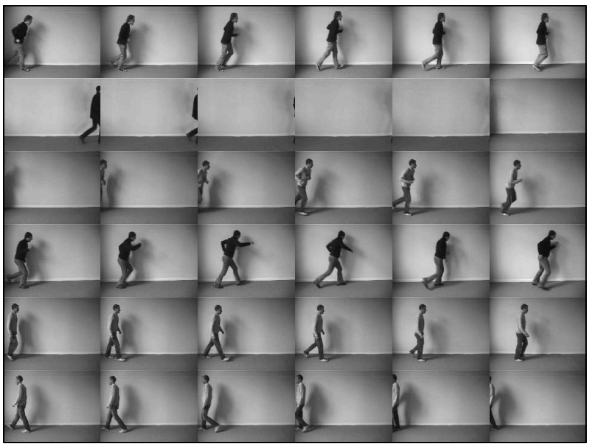
video segment. From the available ground-truth, we see that the elements of the summary correspond to the following activities – Boxing, Boxing, Running, Walking, Hand-clapping, Hand-waving – in the order they appear in Figure 4.6(c). We see that 5 of the 6 activities are captured in the summary. The only activity missing is Jogging. It is important to note that Jogging is very similar to Running and Walking. Hence, the summarization algorithm did not capture Jogging as a distinct activity. For comparison, we show the results of exemplar selection with k-means and N-cut in figures 4.6(a) and 4.6(b). For the k-means algorithm, we ran the algorithm a few times and chose the clustering that had the lowest clustering cost.

4.10.2 Unconstrained SFU Figure Skating video

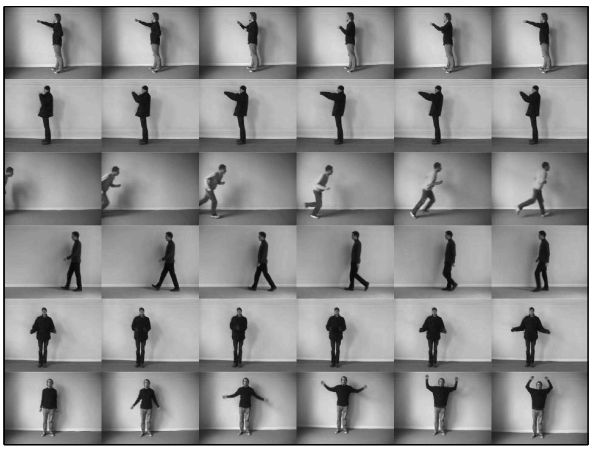
In the next experiment, we used the skating videos from [2] which are completely unconstrained videos with real pan, tilt, and zoom of the camera with rapid motion of the skater. These figure skating videos consist of a few established elements or moves such as jumps, spins, lifts and turns. A typical performance by a skater or a pair of skaters includes several of these elements each performed several times. We show our results on a single skater video which consists of about 3500 frames. The vector representation of each segment is obtained as discussed in 4.9.1. The summary vectors are then chosen using the k-means, N-cut and the proposed algorithm (with $\alpha = 0.45$). Representative frames from each video segment of the generated summary are shown in figures 4.7(a), 4.7(b) and 4.7(c) respectively. Each frame has been marked with the name of the figure skating element being performed



(a) K-Means



(b) N-Cut



(c) Proposed Algorithm

Figure 4.6: Summary of KTH human action database video. Summary of length $K = 6$.

in the corresponding video segment. Note that several segments of the k-means and N-cut summary correspond to 'gliding' (Figures 4.7(a) and 4.7(b)), and it misses out on some of the significant spins, jumps and spirals in the video. On the contrary, the diversity criterion of the proposed algorithm ensures that these varied actions are captured in the summary as shown in Figure 4.7(c).

4.10.3 VIVID surveillance video

In this section, we show the summarization results on DARPA's VIVID dataset. The videos are low resolution aerial videos captured by an Unmanned Aerial Vehicle (UAV). We choose a video of around 7200 frames. To show the complete content of the original video, it has been mosaicked into 3 parts and shown in Figure 4.8(a). Videos in this dataset are continuously captured from a single camera and hence form a single shot-video. Vectorial representation of each video segment (20 frames) is obtained as described in section 4.9.1. $K = 15$ exemplars are chosen as the summary using the N-cut, k-means and the proposed algorithm (with $\alpha = 0.45$). As is evident from the results shown in figures 4.8(b), 4.9(a) and 4.9(b), the summary generated by the proposed algorithm highlights diverse aspects which N-cut or k-means algorithm completely miss out on.

4.10.4 Unconstrained YouTube Video "Office Tour"

In the next experiment, we downloaded a 5 minute home made video "Office Tour" from YouTube. As indicated by the name, this video is about a man touring

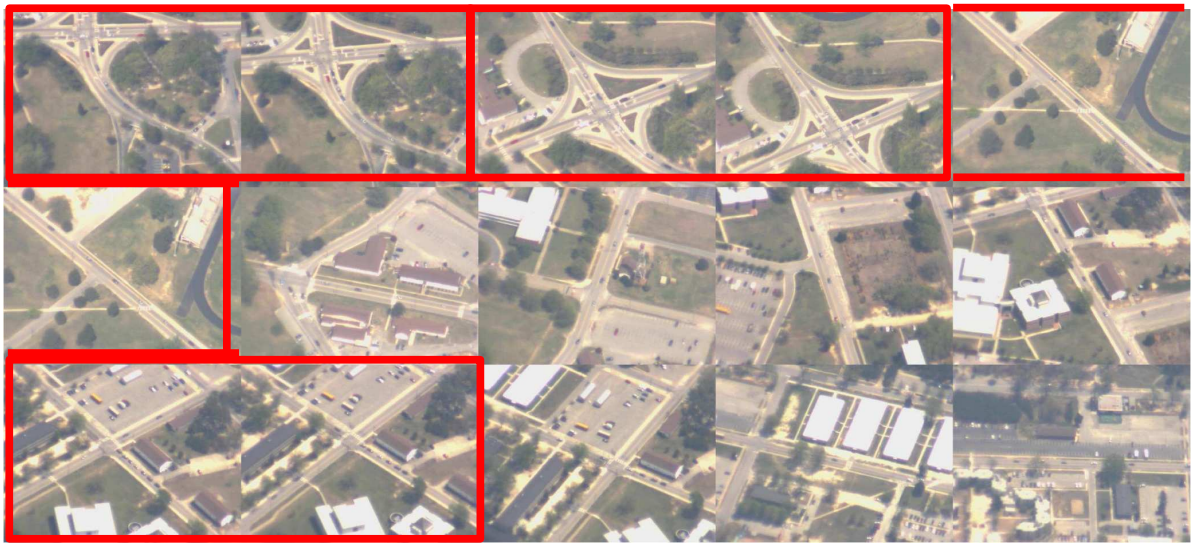
(a) Figure skating video summary using the k-means algorithm.

(b) Figure skating video summary using the N-cut algorithm.

(c) Figure skating video summary using the proposed algorithm.

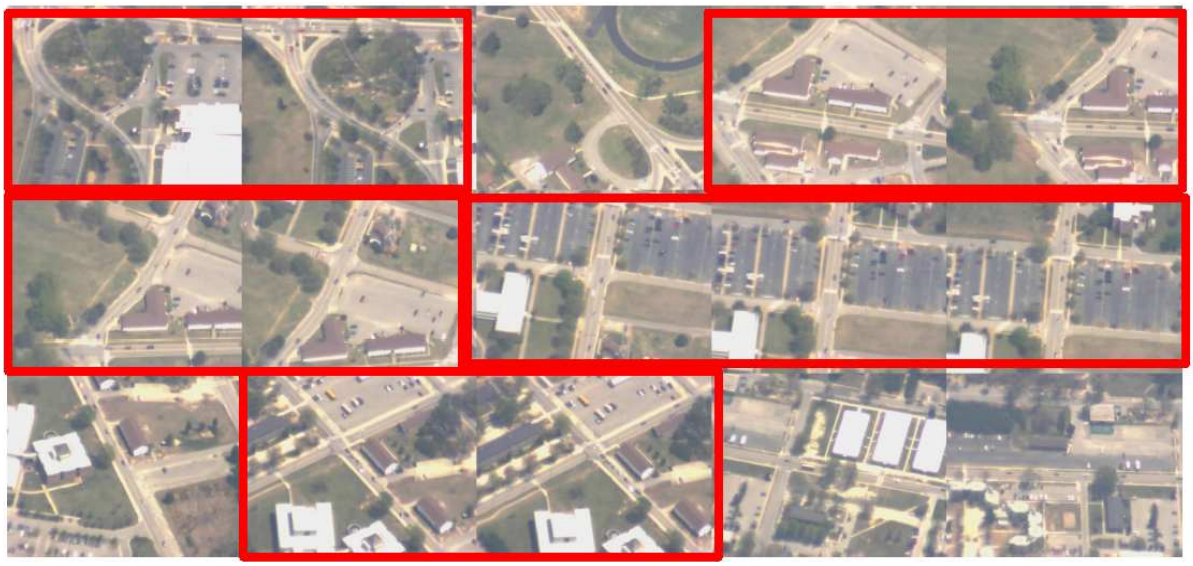
Figure 4.7: Figure skating Video summary ($K = 20$). Both k-means and N-cut selects around 11 exemplars from 'Glide' event, while the proposed algorithm picks more diverse and "interesting" exemplars.

(a) VIVID video mosaicked into 3 parts



(b) VIVID video summary using the N-cut algorithm.

Figure 4.8: (a) VIVID video. Frame of this 7200 frames long video has been mosaicked into 3 parts. These mosaics are shown to give the readers an idea about the full content of the video. (b) Summary generated using N-cut ($K = 15$ length).



(a) VIVID video summary using the k-means algorithm.

(b) VIVID video summary using the proposed algorithm.

Figure 4.9: VIVID video Summary ($K = 15$ length). Shown are the central frames from each video segment of the summary generated. Comparatively, lot many redundant exemplars are chosen by K-means and N-cut.

an office and meeting various employees. Since the video was captured by a handheld camera, it has jitter and other motion artifacts caused due to the hand movements.

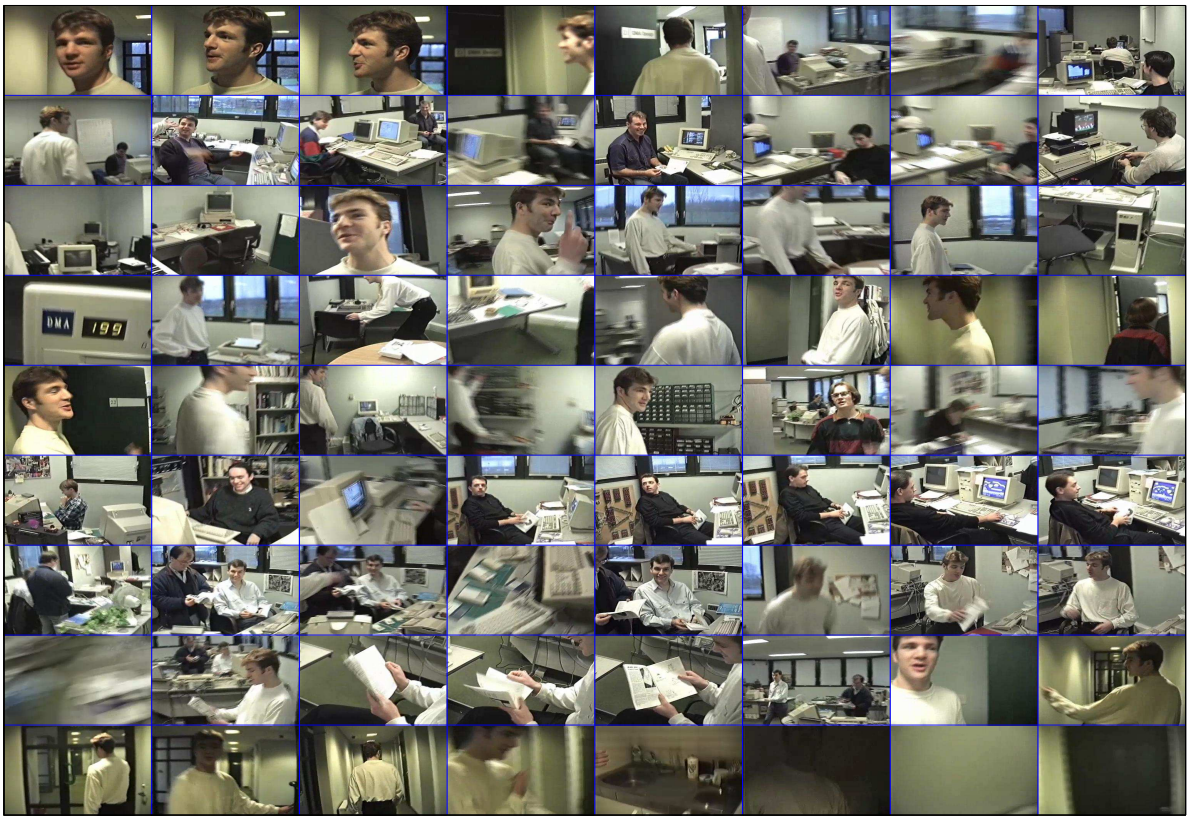
This video was captured in one long shot and had around 7500 frames. This video was chosen to demonstrate the strength of the proposed approach. Vectorial representation of each video segment (20 frames) is obtained as described in section 4.9.1. 72 uniformly sampled frames from the original video are shown in Figure 4.10(a). A summary of 20 exemplars was generated using each of the algorithms (a) N-cut (b) k-means and (c) Proposed algorithm. Central frames from each of the exemplar segments are shown in figures 4.10(b), 4.11(a) and 4.11(b) respectively.

Visualizing the increase in Diversity: To visualize the increase in diversity, we calculate the pairwise distance between the centroids for each of the three unconstrained videos used in our experiments. Larger pairwise distances signify that the centroids are more spread out, leading to greater diversity of the summary. In figures 4.12(a), 4.12(b) and 4.12(c), we show the histograms of pairwise distances between the centroids for (a) Figure Skating (b) VIVID video and (c) YouTube Video respectively. As is evident the proposed algorithm has a larger diversity in all the three videos.

4.11 Qualitative Evaluation

In this section, we show results of a user-study that we conducted to measure the effectiveness of the proposed approach to summarization. Such a study helps in understanding the factors relevant to human perception. We now describe the user-study that we conducted.

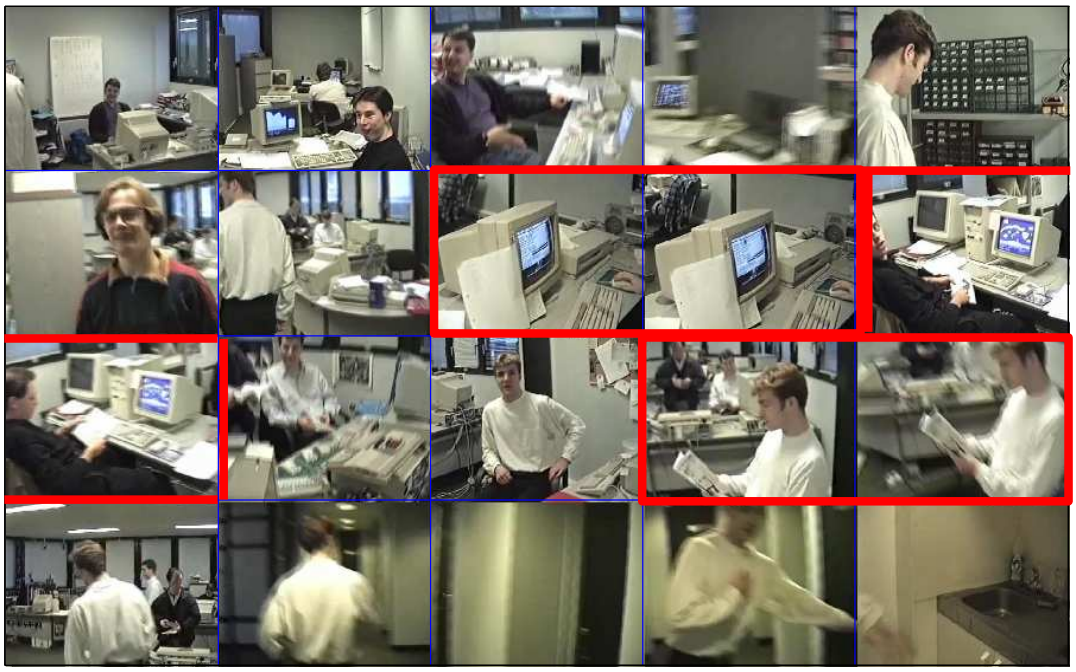
Twelve voluntary subjects took part in the evaluation. None was involved in



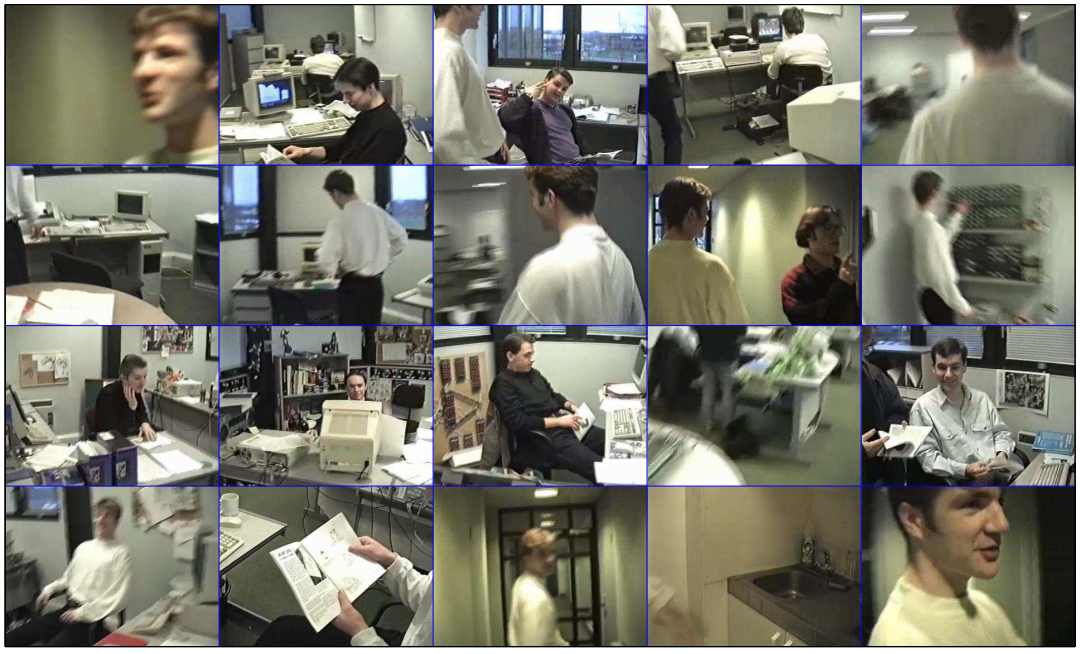
(a) YouTube video uniform samples

(b) YouTube video summary using the N-cut algorithm.

Figure 4.10: YouTube video "Office Tour". This video is around 5 minutes (7500 frames) long. (a) 72 uniformly sampled videos. (b) $K = 20$ length summary generated using N-cut. Shown are the central frames of each video segment in summary. Redundant frames are marked in red boxes.

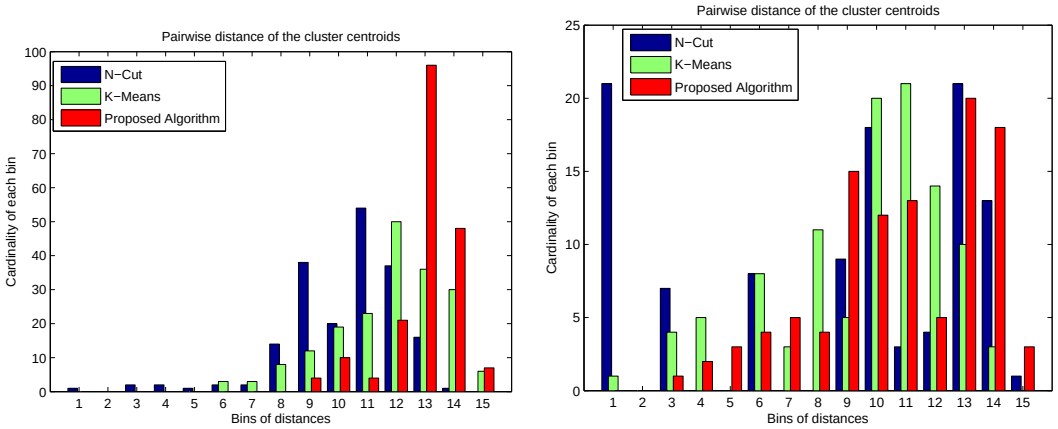


(a) YouTube video summary using the k-means algorithm.

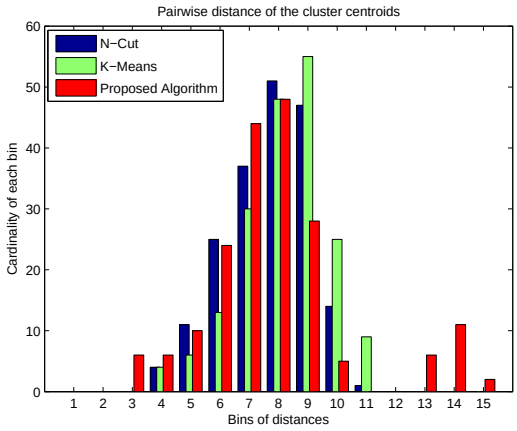


(b) YouTube video summary using the proposed algorithm.

Figure 4.11: YouTube video summary ($K = 20$). Shown are central frames of each video segment of summary. Redundant frames are marked in red boxes.



(a) Figure Skating (b) VIVID video



(c) YouTube video

Figure 4.12: Histogram of pairwise distances of the centroids. Note that the proposed algorithm (red bars) has greater number of points in the higher distance bins which implies that the centroids chosen are more distant than the N-cut (blue bars) and k-means (green bar).

Video	Total frames	Summary Length K	Weight α	# frames per video segment	BoW τ	BoW σ	BoW N
KTH	14400	6	0.45	20	2	2.5	500
Figure Skating	3819	20	0.45	20	2	2.5	500
VIVID video	7200	15	0.45	20	2	2.5	500
Youtube video	7500	20	0.45	20	2	2.5	500

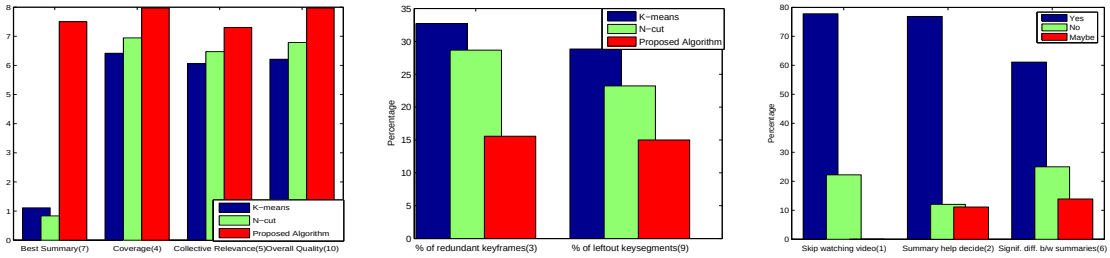
Table 4.1: Parameters chosen for each experiment. BoW here implies bag-of-words discussed in section 4.9.1

the design/implementation of the proposed algorithm, thus were unaware of what to expect. The subjects were allowed to watch the video-sequences – both the originals and the summaries – as many times as they desired in any order they wanted. Breaks were allowed and the subjects could change any of the answers whenever they wanted. First the original video (in the form of full-length video or large number of uniform samples) were shown to the subjects. Then each of the summaries (generated by proposed algorithm, K-means and N-cut - denoted as Algorithms A/B/C) were shown. The algorithm used to generate the summary was *not* revealed to the user but simply labeled as A/B/C. After showing each set of video and the corresponding summaries, users were asked to fill in an evaluation form with the questions and possible answers shown in Table 4.2.

After the evaluations were completed by all the subjects, they were compiled and a few statistics were computed. The results are shown in figures 4.13(a), 4.13(b) and 4.13(c). The first two figures correspond to questions about the comparative

S.No.	Questions	Possible Answer
1	Can you skip watching the video as you already know almost all of the content of the video?	Yes / No
2	Did the summary help you decide whether you would like to watch the original video or not?	Yes / No / May be
3	How many keyframes in each summary were redundant? (Give an approximate percentage).	0 - 100 %
4	Did it capture the essence of the video? i.e., rate the coverage of the video.	1 - 10 (10 is best)
5	Rate the collective relevance of the individual keyframes in each summary.	1 - 10
6	Are there significant differences between the various automated summarization methods?	Yes / No / May be
7	Choose the best summarization scheme.	A / B / C
8	Why do you find it better than other sequences?	Subjective
9	How many important segments do you think were left out? (Give an approximate percentage).	0 - 100 %
10	Overall quality of the summary generated by each scheme.	1 - 10
11	Comments (if any)	Subjective

Table 4.2: Questions used along with their possible answers in the Qualitative evaluation of the summary.



(a) (b) (c)

Figure 4.13: Responses for (a) Survey questions 7, 4, 5 and 10. (b) Survey questions 3 and 9. (c) Survey questions 1, 2 and 6. Numbers in the bracket indicates the corresponding survey question in Table 4.2.

performance between the three algorithms. As can be seen in Figure 4.13(a) the proposed algorithm was given the highest score in all the four questions and hence demonstrates superior quality of the summary. In Figure 4.13(b), as expected the proposed algorithm has the least number of redundant keyframes in its summary and also the least number of important segments left out. Figure 4.13(c) focuses on questions that ask whether the user wants to watch the full-length video after watching the summaries with possible answers of Yes / No / Maybe. Around 80% of users agreed that the summary could help them in deciding if it was interesting to them or not. The subjective answers to survey question number 8 highlighted that the redundancy was reduced by the proposed algorithm and therefore more amount of information was being captured in the summary. This user-based evaluation further supports our conclusions.

4.12 Quantitative Evaluation

We evaluate the summary further, this time quantitatively using a reconstruction error-based cost function. Here, we would like to measure how well the summary centroids can reconstruct the video. To measure this, we define $P(S)$ as the count of the frames whose reconstruction error using the exemplars is above the threshold γ.

$$P(S) = \sum_{i} u\left[(\, min_w \, ||F_i - Sw||^2) - \, \gamma\right] \tag{4.8}$$

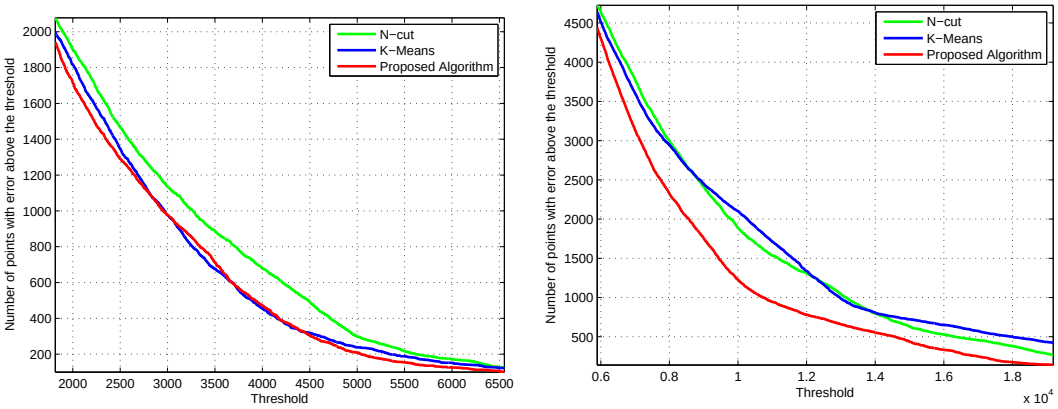
where $S_{d\times K}$ is the matrix of K exemplars each with Euclidean representation in $\mathbb{R}^d$. Here, $u[n]$ is the unit step function, F_i is the d dimensional Euclidean representation of i^{th} frame and γ is the threshold of the reconstruction error above which the frame is counted to be lying in the null space of S. $w_{K\times 1}$ is the vector of the weights assigned to each exemplar. The reconstruction error for each frame is minimized over the w for each F_i which is then chosen to be $w = S^+F_i$, where S^+ is the pseudo-inverse of S.

This cost function supports the intuition about the basic feature of a 'good' summary that all the frames of the video must lie in the space spanned by the linear combination of the exemplars. Therefore, fewer the number of frames in the null space of the exemplars, the better the summary.

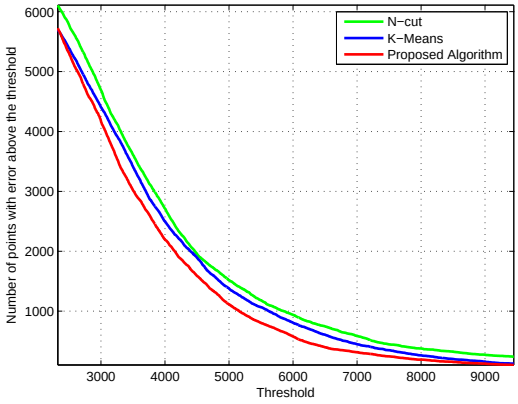
Any evaluation based on the proposed exemplar selection cost function $J(S)$ would have been very much tuned to the choice of features. Hence, the choice of the evaluation cost function $P(S)$ can give importance to the visual aspect of the summary. As this quantitative evaluation $P(S)$ is independent of the proposed

approach to summarization, it makes it usable for evaluation of any summarization algorithm in general.

For the reconstruction each frame was considered as a gray-level intensity image. A histogram of the gray values of each frame and the exemplars was used for the evaluation. This representation further brings out the independence of the evaluation function to the summarization technique method proposed above.



(a) Figure Skating (b) VIVID video



(c) YouTube video

Figure 4.14: Reconstruction error-based objective evaluation and comparison of the (i) N-cut (green), (ii) k-means (blue) and (iii) Proposed algorithm (red).

The results of this evaluation are shown in Figure 4.14. Shown in this plot is the

Video	Total frames	Threshold	N-cut	k-means	Proposed Algorithm
Figure Skating	3819	4.6e3	443	303	281
VIVID video	7200	1.7e4	456	573	248
Youtube video	7500	8492	325	201	154

Table 4.3: Reconstruction Error evaluation. Shown here are the number of frames in the null space of the exemplars chosen by different algorithms for the specified threshold.

number of frames in the null space of the exemplars chosen by different algorithms with varying threshold. Table 4.3 shows the number of frames in the null space at one particular threshold. This evaluation points out that the proposed algorithm has the least number of frames in the null space. The figure skating video has a lot more abrupt changes and dynamics in it as compared to the relatively smoothly changing VIVID video or Youtube video. Therefore, the ratio of frames in the null space is higher as compared to the other two videos. This quantitative evaluation further supports the user-based evaluation in emphasizing the role of the diversity criterion in the video summarization.

4.13 Conclusion and Future Work

We have proposed an unsupervised method for summarizing long videos by quantifying two important criteria, namely – coverage and diversity. We have shown that fairly simple definitions of these concepts translate to significant improvements in summarization quality over more traditional approaches. The improvement has

been demonstrated through experiments on four different class of videos. The summaries thus produced have been evaluated both qualitatively (user based evaluation) and quantitatively (reconstruction error function).

The goal of Video Précis is to provide a condensed and succinct representations of the content of a video. But defining which video segments are 'interesting' is a very subjective process. It is also very difficult to map human cognitive abilities into an automated abstraction process. The difficulty of the problem increases as the properties of a video summary also depend on the application domain, the characteristics of the sequences to be summarized, and the purpose of the summary. We have proposed a method that tries to optimize between the conflicting requirements of coverage and diversity and shown that it is well suited to summarize a large class of unconstrained videos.

Chapter 5

Manifold Précis: An Annealing Technique for Diverse Sampling of Manifolds

5.1 Introduction

The problem of sampling K representative data points from a large dataset arises frequently in various applications. Consider analyzing large datasets of shapes, objects, documents or large video sequences, etc. Analysts spend a large amount of time sifting through the acquired data to familiarize themselves with the content, before using them for their application specific tasks. This has necessitated the problem of optimal selection of a few representative exemplars from the dataset as an important step in exploratory data analysis. Other applications include Internet-based video summarization, where providing a quick overview of a video is important for improving the browsing experience. Similarly, in medical image analysis, picking a subset of K anatomical shapes from a large population helps in identifying the variations within and across shape classes, providing an invaluable tool for analysts.

Depending upon the application, several subset selection criteria have been proposed in the literature. However, there seems to be a consensus in selecting exemplars that are representative of the dataset while minimizing the redundancy between the exemplars. Liu *et al.*[96] proposed that the summary of a document

should satisfy the 'coverage' and 'orthogonality' criteria. Shroff *et al.*[104] extended this idea to selecting exemplars from videos that maximize 'coverage' and 'diversity'. Simon *et al.*[105] formulated scene summarization as one of picking interesting and important scenes with minimal redundancy. Similarly, in statistics, stratified sampling techniques sample the population by dividing the dataset into mutually exclusive and exhaustive 'strata' (sub-groups) followed by a random selection of representatives from each strata [106]. The splitting of population into stratas ensures that a diverse selection is obtained. The need to select diverse subsets has also been emphasized in information retrieval applications [107, 108].

Column Subset Selection (CSS) [109, 110, 111] has been one of the popular techniques to address this problem. The goal of CSS is to select the K most "well-conditioned" columns from the matrix of data points. One of the key assumptions behind this and other techniques is that objects or their representations, lie in the Euclidean space. Unfortunately, this assumption is not valid in many cases. In applications like computer vision, images and videos are represented by features/models like shapes [112], bags-of-words, linear dynamical systems (LDS) [99], etc. Many of these features/models have been shown to lie in non-Euclidean spaces, implying that the underlying distance metric of the space is not the usual ℓ_2/ℓ_p norm. Since these feature/model spaces have a non-trivial manifold structure, the distance metrics are highly non-linear functions. Examples of features/models - manifold pairs include: shapes - complex spherical manifold [112], linear subspaces - Grassmann manifold, covariance matrices - tensor space, histograms - simplex in $\mathbb{R}^n$, etc. Even the familiar bag-of-words representation, used commonly in document analysis, is

more naturally considered as a statistical manifold than as a vector space [113]. The geometric properties of the non-Euclidean manifolds allow one to develop accurate inference and classification algorithms [114, 115]. In this chapter, we focus on the problem of selecting a subset of K exemplars from a dataset of N points when the dataset has an underlying manifold structure. We formulate the notion of representational error and diversity measure of exemplars while utilizing the non-Euclidean structure of the data points followed by the proposal of an efficient annealing-based optimization algorithm.

Related Work: The problem of subset selection has been studied by the communities of numerical linear algebra and theoretical computer science. Most work in the former community is related to the *Rank Revealing* QR factorization (RRQR) [109, 116, 117]. Given a data matrix Y, the goal of RRQR factorization is to find a permutation matrix Π such that the QR factorization of $Y\Pi$ reveals the numerical rank of the matrix. The resultant matrix $Y\Pi$ has as its first K columns the most "well-conditioned" columns of the matrix Y. On the other hand, the latter community has focused on Column Subset Selection (CSS). The goal of CSS is to pick K columns forming a matrix $C \in \mathbb{R}^{m \times K}$ such that the residual

$$|| \, Y - \; P_C Y \, ||_\zeta$$

is minimized over all possible choices for the matrix C. Here $P_C = CC^\dagger$ denotes the projection onto the K-dimensional space spanned by the columns of C and ζ can represent the spectral or Frobenius norm. $C^\dagger$ indicates the pseudo inverse of matrix C. Along these lines, different randomized algorithms have been proposed [118,

119, 111, 110]. Various approaches include a two-stage approach [111], subspace sampling methods [110], etc.

Clustering techniques [120] have also been applied for subset selection [121, 101]. In order to select K exemplars, data points are clustered into ℓ clusters with $(\ell \leq K)$ followed by the selection of one or multiple exemplars from each cluster to obtain the best representation or low-rank approximation of each cluster. Affinity Propagation [101], is a clustering algorithm that takes similarity measures as input and recursively passes message between nodes until a set of exemplars emerges. As we discuss in this chapter, the problems with these approaches are that (a) the objective functions optimized by the clustering functions do not incorporate the diversity of the exemplars, hence can be biased towards denser clusters, and also by outliers, and (b) seeking low-rank approximation of the data matrix or clusters individually is not always an appropriate subset selection criterion. Furthermore, these techniques are largely tuned towards addressing the problem in an Euclidean setting and cannot be applied for datasets in non-Euclidean spaces.

Recently, advances have been made in utilizing non-Euclidean structure for statistical inferences and pattern recognition [114, 115, 122, 123]. These works have addressed inferences, clustering, dimensionality reduction, etc. in non-Euclidean spaces. To the best of our knowledge, the problem of subset selection for analytic manifolds remains largely unaddressed. While one could try to solve the problem by obtaining an embedding of a given manifold into a larger ambient Euclidean space, it is desirable to have a solution that is more intrinsic in nature. This is because the chosen embedding is often arbitrary, and introduces peculiarities that result

from such extrinsic approaches. Further manifolds such as the Grassmannian or the manifold of infinite dimensional diffeomorphisms do not admit a natural embedding into a vector space.

Contributions: In this chapter, we make the following contributions:

1. We present the first formal treatment of subset selection for the general case of manifolds.

2. We propose a novel annealing-based alternation algorithm to efficiently solve the optimization problem.

3. We present an extension of the algorithm for data manifolds, and demonstrate the favorable properties of the algorithm on real data.

5.2 Examples of Non-linear manifolds

We will discuss a few manifolds that frequently appear in vision applications.

1. **Shape Features:** Shapes in images are commonly described by a set of landmarks extracted from the object being imaged. After appropriate translation, scale and rotation normalization it can be shown that shapes reside on a complex spherical manifold [112]. Further, by factoring out all possible affine transformations, it can be shown that shapes reside on a Grassmann manifold. More recently, shapes have been considered to be continuous closed planar curves. The space of such curves can also be characterized as a manifold [124].

2. **Covariance Features:** In recent years, region covariance has proved to be a popular feature to encode local shape and texture. Covariance features as region descriptors were introduced in [100] and have been successfully applied to human detection [125], object tracking [126] and texture classification [100]. Covariance matrices also appear in medical imaging literature where diffusion tensor MRI produces measurements of diffusion of water molecules, where each voxel is associated with a 3×3 symmetric positive definite matrix [127].

3. **Time Warps:** The space of positive and monotonically increasing functions mapping the unit-interval to the unit-interval are usually referred to as time-warp functions. The derivatives of warping functions can be interpreted as probability density functions. The square-root form of pdfs can then be described as a sphere in the space of functions. This was exploited in [128] to recognize human activities. Variability in sampling closed planar curves can also be modeled as a sphere in the space of functions [115].

4. **Subspaces:** In image and object recognition, recent methods have focused on utilizing multiple images of the same object, taken under varying viewpoints or varying illumination conditions, for recognition [129, 130, 131, 132]. The set of face images of the same person under varying illumination conditions is frequently modeled as a linear subspace of 9-dimensions [133]. The set of k-dimensional subspaces of $\mathbb{R}^n$ is called the Grassmann manifold. A related manifold of k-basis vectors of $\mathbb{R}^n$ or $n \times k$ tall-thin orthonormal matrices is called the Stiefel manifold and has found applications in face recognition [134].

The Stiefel manifold also finds applications in finding projections of data with desired statistical properties such as sparsity, discriminability, etc [135].

5. **Dynamic models:** In video analysis, an important task is to describe a sequence of static features using parametric models. One popular dynamical model for such time-series data is the autoregressive and moving average (ARMA) model. Examples include dynamic textures [99], human joint angle trajectories and silhouette sequences [136]. The space spanned by the columns of the observability matrix of the ARMA model can be identified as a point on the Grassmann manifold [137].

5.3 Preliminaries

In this section, we briefly recapitulate the mathematical preliminaries needed to study nearest-neighbor searching on non-Euclidean manifolds. A topological space $\mathcal{M}$ is called a manifold if it is *locally Euclidean* i.e. for each $p \in \mathcal{M}$, there exists an open neighborhood U of p and a mapping $\phi : U \to \mathbb{R}^n$ such that $\phi(U)$ is open in $\mathbb{R}^n$ and $\phi : U \to \phi(U)$ is a diffeomorphism. The pair (U, ϕ) is called a *coordinate chart* for the points that fall in U. Let $\mathcal{M}$ be an n-dimensional manifold and, for a point $p \in \mathcal{M}$, consider a differentiable curve $\gamma : (-\epsilon, \epsilon) \to \mathcal{M}$ such that $\gamma(0) = p$. The velocity $\dot{\gamma}(0)$ denotes the velocity of γ at p. This vector has the same dimension as the manifold and is an example of a tangent vector to $\mathcal{M}$ at p. The set of all such tangent vectors is called the tangent space to $\mathcal{M}$ at p. Even though the manifold $\mathcal{M}$ maybe nonlinear, the tangent space $T_p(\mathcal{M})$ is always linear.

5.3.1 Metrics via Geodesic Distances

The task of measuring distances on a manifold is accomplished using a Riemannian metric. A Riemannian metric on a differentiable manifold $\mathcal{M}$ is a map $\langle \cdot, \cdot \rangle$ that smoothly associates to each point $p \in \mathcal{M}$ a symmetric, bilinear, positive definite form on the tangent space $T_p(\mathcal{M})$. Using the Riemannian structure, it becomes possible to define lengths of paths on a manifold. For any two points $p, q \in \mathcal{M}$, one can define the distance between them as the infimum of the lengths of all smooth paths on $\mathcal{M}$ which start at p and end at q:

$$d(p,q) = \inf_{\{\alpha:[0,1]\mapsto\mathcal{M}|\alpha(0)=p,\alpha(1)=q\}} L[\alpha], \quad \text{where}$$

$$L[\alpha] = \int_0^1 \sqrt{\left(\left\langle \frac{d\alpha(t)}{dt}, \frac{d\alpha(t)}{dt} \right\rangle\right)} dt$$

5.3.1.1 Tangent-plane embedding

If $\mathcal{M}$ is a Riemannian manifold and $p \in \mathcal{M}$, the exponential map $\exp_p : T_p(\mathcal{M}) \to \mathcal{M}$, is defined by $\exp_p(v) = \alpha_v(1)$ where α_v is a specific geodesic. The inverse mapping $\log_p : \mathcal{M} \to T_p$ called the inverse exponential map/logarthmic map at a 'pole', takes a point on the manifold and returns a point on the tangent space of the pole – which is a Euclidean space.

5.3.1.2 Centroids

Given a set of points on a manifold, the intrinsic mean of the dataset or the Karcher mean [138] is a natural way of generalizing the notion of a centroid. The

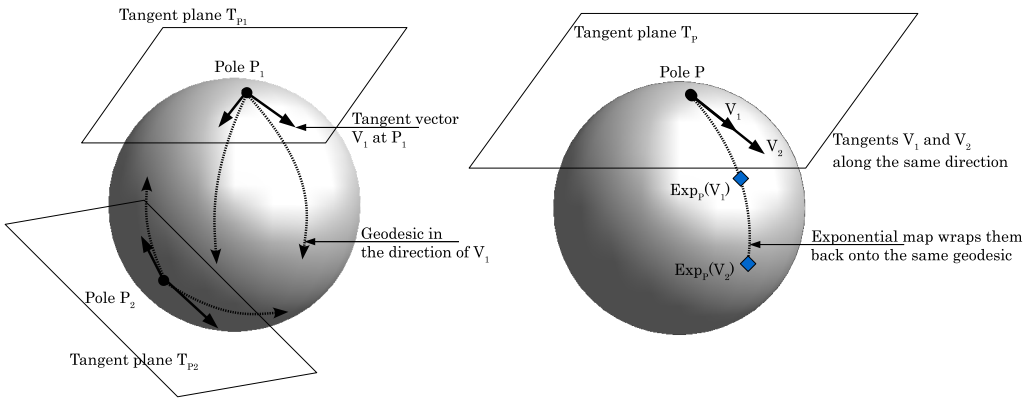


Figure 5.1: (Left) Figure illustrating the notions of tangent spaces, tangent vectors, and geodesics. Shown in the figure are two points P_1 and P_2 on the manifold and the tangent planes at these points. Geodesics paths are constant velocity curves on the manifold. Tangent vectors correspond to velocities of curves on the manifold. (Right) Figure illustrating the notion of exponential maps and inverse exponential maps. Shown are two points on the tangent plane at pole P. Both points lie along the same tangent vector. The exponential map will map them onto the same geodesic.

intrinsic mean is defined as the point μ that minimizes the sum of squared-distance to all other points

$$\mu = arg \min_{x \in M} \sum_{i=1}^{N} d(x, x_i)^2$$

Computing the intrinsic mean is not usually possible in a closed form. The intrinsic mean is unique only for points that are close together [138]. An iterative procedure is used for estimation of means of points on manifolds [139]. These concepts are illustrated graphically in Figure 5.1.

Symbol	Represents
Γ	Maximum number of iterations
$\mathcal{I}_{n\times n}$	Identity matrix
Φ_i	Voronoi region of i^{th} exemplar
$\Pi_{i\leftrightarrow j}$	Permutation matrix that swaps columns i and j
$\Pi^{(\gamma)}$	Π in the γ^{th} iteration
V	Matrix obtained by tangent-space projection of X
H_{ij}	(i,j) element of matrix H
α_j	j^{th} column of the identity matrix
$H\alpha_j, \alpha_j^T H$	j^{th} column and row of matrix H respectively

Table 5.1: Notations used in Algorithms 5.1 - 5.3

5.4 Subset Selection on Analytic Manifolds

In this section, we formalize the subset selection problem on manifolds and propose an efficient algorithm.

5.4.1 Representational error on manifolds

Let us assume that we are given a set of points $X = \{x_1, x_2, \ldots x_n\}$ which belong to a manifold $\mathcal{M}$. The goal is to select a few exemplars $E = \{e_1, \ldots e_K\}$ from the set X, such that the exemplars provide a good representation of the given data points, and are minimally redundant. For the special case of vector spaces, two common approaches for measuring representational error is in terms of linear spans, and nearest-exemplar error. The linear span error is given by

$$\min_z \|\mathbf{X} - \mathbf{E}z\|_F^2$$

where, $\mathbf{X}$ is the matrix form of the data, and $\mathbf{E}$ is a matrix of chosen exemplars. The nearest-exemplar error is given by

$$\sum_i \sum_{x_k \in \Phi_i} \|x_k - e_i\|^2$$

where e_i is the i^{th} exemplar and Φ_i corresponds to its Voronoi region.

Of these two measures, the notion of linear span, while appropriate for matrix approximation, is not particularly meaningful for general dataset approximation problems since the 'span' of a dataset item does not carry perceptually meaningful information. For example, the linear span of a vector $x \in \mathbb{R}^n$ is the set of points $\alpha x, \alpha \in \mathbb{R}$. However, if x were an image, the linear span of x would be the set of

images obtained by varying the global contrast level. All elements of this set however are perceptually equivalent, and one does not obtain any representational advantage from considering the span of x. Further, points sampled from the linear span of few images, would not be meaningful images. This situation is further complicated for manifold-valued data such as shapes, where the notion of linear span does not exist. One could attempt to define the notion of linear spans on the manifold as the set of points lying on the geodesic shot from some fixed pole toward the given dataset item. But, points sampled from this linear span might not be very meaningful e.g., samples from the linear span of a few shapes would give physically meaningless shapes.

Hence, it is but natural to consider the representational error of a set X with respect to a set of exemplars E as follows:

$$J_{rep}(E) = \sum_{i} \sum_{x_j \in \Phi_i} d_g^2(x_j, e_i) \tag{5.1}$$

Here, d_g is the geodesic distance on the manifold and Φ_i is the Voronoi region of the i^{th} exemplar. This boils down to the familiar K-means or K-medoids cost function for Euclidean spaces. In order to avoid combinatorial optimization involved in solving this problem, we use efficient approximations i.e., we first find the mean followed by the selection of e_i as data point that is closest to the mean. The algorithm for optimizing J_{rep} is given in Algorithm 5.1. Similar to K-means clustering, a cluster label is assigned to each x_j followed by the computation of the mean μ_i for each cluster. This is further followed by selecting representative exemplar e_i as

the data point closest to μ_i.

Algorithm 5.1: Algorithm to minimize J_{rep}

Input: $X \in \mathcal{M}$, k, index vector ω, Γ

Output: Permutation Matrix Π

Initialize $\Pi \leftarrow \mathcal{I}_{n\times n}$

for $\gamma \leftarrow 1$ **to** Γ **do**

Initialize $\Pi^{(\gamma)} \leftarrow \mathcal{I}_{n\times n}$

$e_i \leftarrow x_{\omega_i}$ for $i = \{1,2,\ldots,k\}$

for $i \leftarrow 1$ **to** k **do**

$\Phi_i \leftarrow \{x_p : arg\ min_j\ d_g(x_p, e_j) = i\ \}$

$\mu_i \leftarrow$ mean of Φ_i

$\hat{j} \leftarrow arg\ min_j\ d_g(x_j, \mu_i)$

Update: $\Pi^{(\gamma)} \leftarrow \Pi^{(\gamma)}\ \Pi_{i\leftrightarrow\hat{j}}$

end

Update: $\Pi \leftarrow \Pi\ \Pi^{(\gamma)}$, $\omega \leftarrow \omega\Pi^{(\gamma)}$

if $\Pi^{(\gamma)} = \mathcal{I}_{n\times n}$ **then**

break

end

end

5.4.2 Diversity measures on manifolds

The next question we consider is to define the notion of diversity of a selection of points on a manifold. We first begin by examining equivalent constructions for

$\mathbb{R}^n$. One of the ways to measure diversity is simply to use the sample variance of the points. This is similar to the construction used recently in [104]. For the case of manifolds, the sample variance can be replaced by the sample Karcher variance, given by the function:

$$\rho(E) = \frac{1}{K}\sum_{i=1}^{K} d_g^2(\mu, e_i)$$

where μ is the Karcher mean [138], and the function value is the Karcher variance. However, this construction leads to highly inefficient optimization routines, essentially boiling down to a combinatorial search over all possible K-sized subsets of X.

An alternate formulation for vector spaces that results in highly efficient optimization routines is via Rank-Revealing QR (RRQR) factorizations. For vector spaces, given a set of vectors $\mathbf{X} = \{x_i\}$, written in matrix form $\mathbf{X}$, RRQR [109] aims to find Q, R and a permutation matrix $\Pi \in \mathbb{R}^{n\times n}$ such that $\mathbf{X}\Pi = QR$ reveals the numerical rank of the matrix $\mathbf{X}$. This permutation

$$\mathbf{X}\Pi = (\mathbf{X}_K \;\; \mathbf{X}_{n-K})$$

gives $\mathbf{X}_K$, the K most linearly independent columns of $\mathbf{X}$. This factorization is achieved by seeking Π which maximizes

$$\Lambda(\mathbf{X}_K) = \prod_i \sigma_i(\mathbf{X}_K)$$

the product of the singular values of the matrix $\mathbf{X}_K$.

For the case of manifolds, we adopt an approximate approach in order to measure diversity in terms of the 'well-conditioned' nature of the set of exemplars

projected on the tangent space at the mean. In particular, for the dataset $\{x_i\} \subseteq \mathcal{M}$, with intrinsic mean μ, and a given selection of exemplars $\{e_j\}$, we measure the diversity of exemplars as follows: matrix $\mathbf{T}_E = [\log_\mu(e_j)]$ is obtained by projecting the exemplars $\{e_j\}$ on the tangent space at mean μ. Here, log() is the inverse exponential map on the manifold and gives tangent vectors at μ that point towards e_j. Diversity can then be quantified as

$$J_{div}(E) = \Lambda(\mathbf{T}_E)$$

where, $\Lambda(\mathbf{T}_E)$ represents the product of the singular values of the matrix $\mathbf{T}_E$. For vector spaces, this measure is related to the sample variance of the chosen exemplars. For manifolds, this measure is related to the sample Karcher *variance*. If we denote $\mathbf{T}_X = [\log_\mu(x_i)]$, the matrix of tangent vectors corresponding to all data-points, and if Π is the permutation matrix that orders the columns such that the first K columns of $\mathbf{T}_X$ correspond to the most diverse selection, then

$$J_{div}(E) = \Lambda(\mathbf{T}_E) = det(R_{11}), \text{where, } \mathbf{T}_X\Pi = QR = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \tag{5.2}$$

Here, $R_{11} \in \mathbb{R}^{K\times K}$ is the upper triangular matrix of $R \in \mathbb{R}^{n\times n}$, $R_{12} \in \mathbb{R}^{K\times(n-K)}$ and $R_{22} \in \mathbb{R}^{(n-K)\times(n-K)}$. The advantage of viewing the required quantity as the determinant of a sub-matrix on the right hand-side of the above equation is that one can obtain efficient techniques for optimizing this cost function. The algorithm for optimizing J_{div} is adopted from [109] and described in Algorithm 5.2. Input to the algorithm is a matrix V created by the tangent-space projection of X

and output is the K most "well-conditioned" columns of V. This is achieved by first decomposing V into QR and computing β_{ij}, which indicates the benefit of swapping i^{th} and j^{th} columns [109]. Algorithm 5.3 then selects pair $(\hat{i}, \hat{j})$ corresponding to the maximum benefit swap β_m and if $\beta_m > tol$, this swap is accepted. This is repeated until either $\beta_m < tol$ or maximum number of iterations is completed.

Algorithm 5.2: Algorithm to maximize J_{div}

Input: Matrix $V \in \mathbb{R}^{d \times n}$, k, Tolerance tol

Output: Permutation Matrix Π

Initialize $\Pi \leftarrow \mathcal{I}_{n \times n}$

repeat

Compute QR decomposition of V to obtain R_{11}, R_{12} and R_{22} s.t., $V = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$

$\beta_{ij} \leftarrow \sqrt{(R_{11}^{-1} R_{12})_{ij}^2 + ||R_{22}\alpha_j||_2^2 ||\alpha_i^T R_{11}^{-1}||_2^2}$

$\beta_m \leftarrow \max_{ij} \beta_{ij}$

$(\hat{i}, \hat{j}) \leftarrow arg\ max_{ij} \beta_{ij}$

Update: $\Pi \leftarrow \Pi\, \Pi_{i \leftrightarrow (j+k)}$

$V \leftarrow V \Pi_{i \leftrightarrow (j+k)}$

until $\beta_m < tol$;

5.4.3 Representation and Diversity Trade-offs for Subset Selection

From (5.1) and (5.2), it can be seen that we seek a solution that represents a trade-off between two conflicting criteria. As an example, in Figure 5.2(a) we show two cases, where J_{rep} and J_{div} are individually optimized. We can see that the

solutions look quite different in each case. One way to write the global cost function is as a weighted combination of the two. However, such a formulation does not lend itself to efficient optimization routines (c.f. [104]). Further, the choice of weights is often left unjustified. Instead, we propose an annealing-based alternating technique of optimizing the conflicting criteria J_{rep} and J_{div}. Optimization algorithms for J_{rep} and J_{div} individually are given in Algorithms 5.1 and 5.2 respectively. We first optimize J_{div} to obtain an initial set of exemplars, and use this set as an initialization for optimizing J_{rep}. The output of this stage is used as the current solution to further optimize J_{div}. However, with each iteration, we increase the tolerance parameter *tol* in Algorithm 5.2. This has the effect of accepting only those permutations that increase the diversity by a higher factor as iterations progress. This is done to ensure that the algorithm is guided towards convergence. If the *tol* value is not increased at each iteration, then optimizing J_{div} will continue to provide a new solution at each iteration that modifies the cost function only marginally. This is illustrated in Ffigure 5.2(c), where we show how the cost functions J_{rep} and J_{div} exhibit an oscillatory behavior if annealing is not used.

As seen in Figure 5.2(b) , the convergence of J_{div} and J_{rep} is obtained very quickly on using the proposed annealing alternation technique. The complete annealing based alternation algorithm is described in Algorithm 5.3. A technical detail to be noted here is that for Algorithm 5.2, input matrix $V \in \mathbb{R}^{d \times n}$ should have $d \geq k$. For cases where $d < k$, Algorithm 5.2 can be replaced by its extension proposed in [111]. Table 5.1 shows the notations introduced in Algorithms 5.1 - 5.3. $\Pi_{i \leftrightarrow j}$ is obtained by permuting i and j columns of the identity matrix.

Algorithm 5.3: Annealing-based Alternation Algorithm for Subset Selection on Manifolds

Input: Data points $X = \{x_1, x_2, \ldots, x_n\} \in \mathcal{M}$, Number of exemplars k,

Tolerance step δ

Output: $E = \{e_1, \ldots e_k\} \subseteq X$

Initial setup:

Compute intrinsic mean μ of X

Compute tangent vectors $v_i \leftarrow \log_\mu(x_i)$

$V \leftarrow [v_1, v_2, \ldots, v_n]$

$\omega \leftarrow [1, 2, \ldots, n]$ be the $1 \times n$ index vector of X

Tol $\leftarrow 1$

Initialize: $\Pi \leftarrow$ Randomly permute columns of $\mathcal{I}_{n\times n}$

Update: $V \leftarrow V\Pi, \omega \leftarrow \omega\Pi$.

while $\Pi \neq \mathcal{I}_{n\times n}$ **do**

Diversity: $\Pi \leftarrow \text{Div}(V, k, tol)$ as in Algorithm 5.2.

Update: $V \leftarrow V\Pi$, $\omega \leftarrow \omega\Pi$.

Representative Error: $\Pi \leftarrow \text{Rep}(X, k, \omega, 1)$ as in Algorithm 5.1

Update: $V \leftarrow V\Pi$, $\omega \leftarrow \omega\Pi$.

$tol \leftarrow tol + \delta$

end

$e_i \leftarrow x_{\omega_i}$ for $i = \{1,2,\ldots,k\}$

5.5 Complexity, Special cases and Limitations

In this section, we discuss how the proposed method relates to the special case of $\mathcal{M} = \mathbb{R}^n$, and to sub-manifolds of $\mathbb{R}^n$ specified by a large number of samples. For the case of $\mathbb{R}^n$, the cost functions J_{rep} and J_{div} boil down to the familiar notions of clustering and low-rank matrix approximation respectively. In this case, Algorithm 5.3 reduces to alternation between clustering and matrix approximation with the annealing ensuring that the algorithm converges. This results in a new algorithm for subset-selection in vector spaces.

For the case of manifolds implicitly specified using samples, one can approach the problem in one of two ways. The first would be to obtain an embedding of the space into a Euclidean space and apply the special case of the algorithm for $\mathcal{M} = \mathbb{R}^n$. The embedding here needs to preserve the geodesic distances between all pairs of points. Multi-dimensional scaling can be used for this purpose. However, recent methods have also focused on estimating logarithmic maps numerically from sampled data points [140]. This would make the algorithm directly applicable for such cases, without the need for a separate embedding. Thus the proposed formalism can accommodate manifolds with known and unknown geometries.

However, the formalism is limited to manifolds of finite dimension. The case of infinite dimensional manifolds, such as diffeomorphisms [141], space of closed curves [142], etc. pose problems in formulating the diversity cost function. While J_{div} could have been framed purely in terms of pairwise geodesics, making it extensible to infinite dimensional manifolds, it would have made the optimization a significant

bottleneck, as already discussed in section 5.4.

5.5.1 Computational Complexity

The computational complexity of computing exponential map and its inverse is specific to each manifold. Let n be the number of data points and K be the number of exemplars to be selected. Table 5.2 enumerates the complexity of different computational step of the algorithm. The last two rows show the complexity of an efficient algorithm proposed by [143] to compute the exponential map and its inverse for the case of Grassmann manifold $\mathcal{G}_{m,p}$.

5.6 Experiments

Baselines: We compare the proposed algorithm with two baselines. The first baseline is a clustering-based solution to subset selection, where we cluster the dataset into K clusters, and pick as exemplar the data point that is closest to the cluster centroid. Since clustering optimizes only the representation cost-function, we do not expect it to have the diversity of the proposed algorithm. This corresponds to the special case of optimizing only J_{rep}. The second baseline is to apply a tangent-space approximation to the entire data-set at the mean of the dataset, and then apply a subset-selection algorithm such as RRQR. This corresponds to optimizing only J_{div} where the input matrix is the matrix of tangent vectors. Since minimization of J_{rep} is not explicitly enforced, we do not expect the exemplars to be the best representatives, even though the set is diverse.

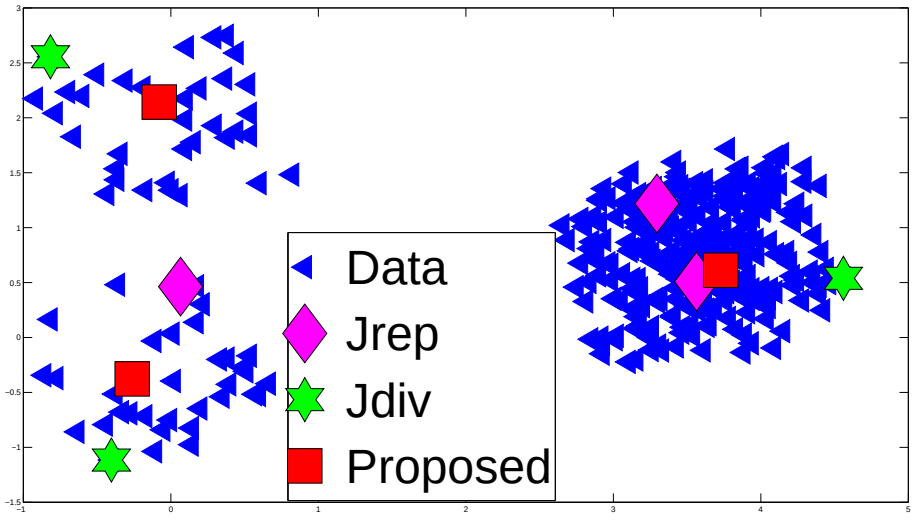
Computational Step	Complexity
$\mathcal{M}$ Exponential Map (assume)	$O(\nu)$
$\mathcal{M}$ Inverse exponential Map (assume)	$O(\chi)$
Intrinsic mean of X	$O((n\chi + \nu)\Gamma)$
Projection of X to tangent-space	$O(n\chi)$
Geodesic distances in Algorithm 5.1	$O(nK\chi)$
K intrinsic means	$O((n\chi + K\nu)\Gamma)$
Algorithm 5.2	$O(mnK \log n)$
$\mathcal{G}_{m,p}$ Exponential Map	$O(p^3)$
$\mathcal{G}_{m,p}$ Inverse exponential map	$O(p^3)$

Table 5.2: Complexity of various computational steps.

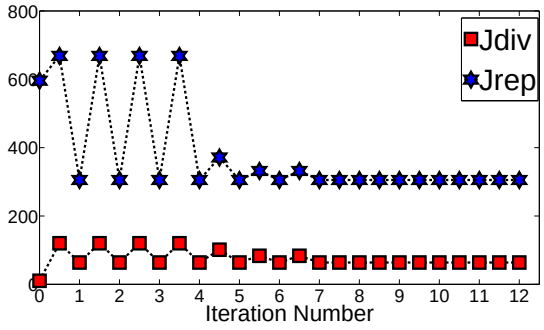
5.6.1 A Simple Dataset

To gain some intuition, we first perform experiments on a simple synthetic dataset. For easy visualization and understanding, we generated a dataset with 3 unbalanced classes in Euclidean space $\mathbb{R}^4$. Individual cost functions, J_{rep} and J_{div} were first optimized to pick three exemplars using algorithms 5.1 and 5.2 respectively. Selected exemplars have been shown in figure 5.2(a), where the four dimensional dataset has been projected into two dimensions for visualization using Principal Component Analysis (PCA). Despite unbalanced class sizes, when optimized individually, J_{div} seeks to select exemplars from diverse classes but tends to pick them from the class boundaries. While unbalanced class sizes cause J_{rep} to pick 2 exemplars from the dominant cluster. Algorithm 5.3 iteratively optimizes for both these cost functions and picks an exemplar from every class. These exemplars, are closer to the centroid of the individual classes.

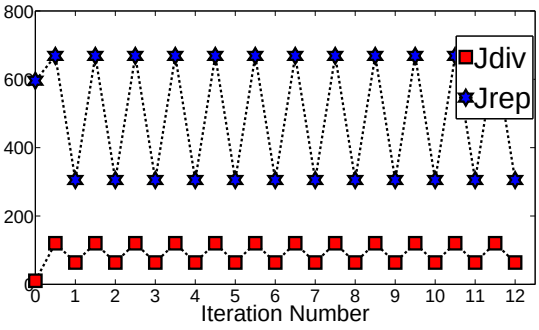
Figure 5.2(b) shows the convergence of the algorithm for this simple dataset and compares it with the case when no annealing is applied (figure 5.2(c)). J_{rep} and J_{div} plots are shown as the iterations of Algorithm 5.3 progresses. When annealing is applied, the tolerance value (tol) is increased by 0.05 in each iteration. It can be noted that in this case the algorithm converges to a steady state in 7 iterations ($tol = 1.35$). If no annealing is applied, the algorithm does not converge.



(a) Subset Selection



(b) Convergence With Annealing



(c) Without Annealing

Figure 5.2: Subset selection for a simple dataset consisting of unbalanced classes in $\mathbb{R}^4$. (a) Data projected on $\mathbb{R}^2$ for visualization using PCA. While trying to minimize the representational error, J_{rep} picks two exemplars from the dominant class. J_{div} picks diverse exemplars but from the boundaries. The proposed approach strikes a balance between the two and picks one 'representative' exemplar from each class. Convergence Analysis of Algorithm 5.3: (b) with annealing and (c) without annealing.

5.6.2 Shape sampling/summarization

We conducted a real shape summarization experiment on the MPEG dataset [144]. This dataset has 70 shape classes with 20 shapes per class. For our experiments, we created a smaller dataset of 10 shape classes with 10 shapes per class. Figure 5.3 shows the shapes used in our experiments. We use an affine-invariant representation of shapes based on landmarks. Shape boundaries are uniformly sampled to obtain m landmark points. These points are concatenated in a matrix to obtain the landmark matrix $\mathcal{L} \in \mathbb{R}^{m\times 2}$. Left singular vectors ($U_{m\times 2}$), obtained by the singular value decomposition of matrix $\mathcal{L} = U\Sigma V^T$, give the affine-invariant representation of shapes [145]. This affine shape-space U of m landmark points is a 2-dimensional subspace of $\mathbb{R}^m$. These p-dimensional subspaces in $\mathbb{R}^m$ constitute the Grassmann manifold $\mathcal{G}_{m,p}$. Details of the algorithms for the computation of exponential and inverse exponential map on $\mathcal{G}_{m,p}$ can be found in [143] and has also been included in appendix B.

Cardinality of the subset in the experiment was set to 10. As the number of shape classes is also 10, one would ideally seek one exemplar from each class. Algorithms 5.1 and 5.2 were first individually optimized to select the optimal subset. Algorithm 5.1 was applied intrinsically on the manifold with multiple initializations. Figure 5.4(a) shows the output with the least cost among these initializations. For Algorithm 5.2, data points were projected on the tangent space at the mean using the inverse exponential map and the selected subset is shown in figure 5.4(b). Individual optimization of J_{rep} results in 1 exemplar each from 6 classes, 2 each from 2 classes

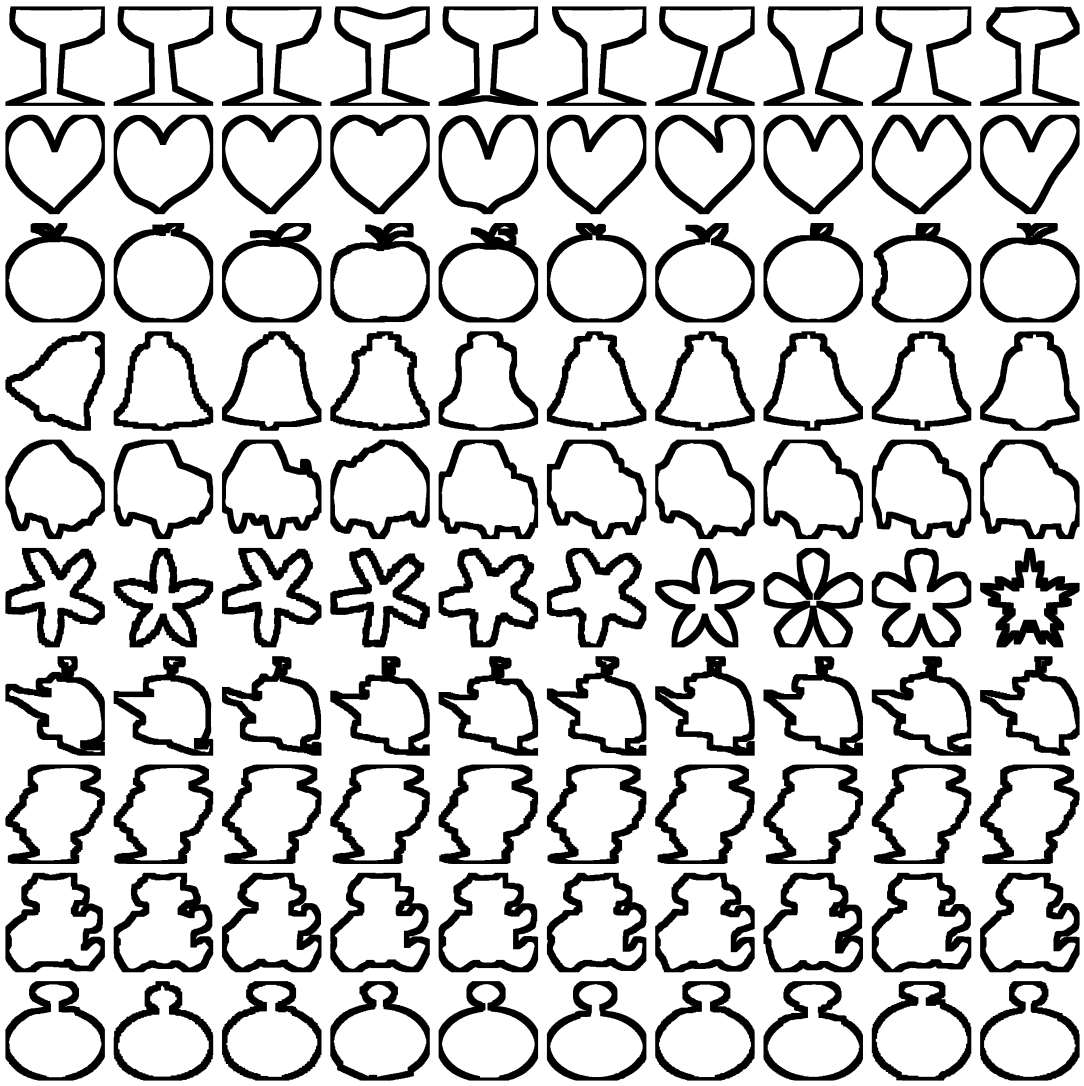
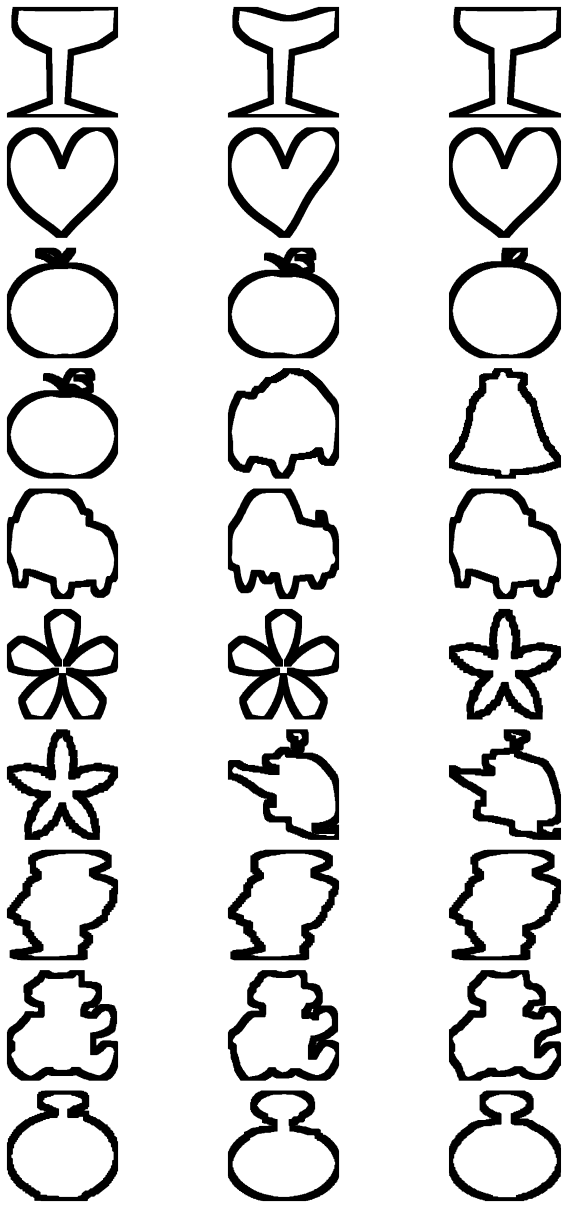


Figure 5.3: 10 classes from MPEG dataset with 10 shapes per class.



(a) (b) (c)

Figure 5.4: Comparison of 10 exemplars selected by (a)J_{rep}, (b) J_{div} and (c) Proposed approach. J_{rep} picks 2 exemplars each from 2 classes ('apple' and 'flower') and misses 'bell' and 'chopper' classes. J_{div} picks 1 from 8 different classes, 2 exemplars from class 'car' and none from class 'bell'. It can be observed that exemplars chosen by J_{div} for classes 'glass', 'heart', 'flower' and 'apple' tend to be unusual members of the class. It also picks up the flipped car. While the proposed approach picks one representative exemplars from each class as desired.

('apple' and 'flower') and misses 2 classes ('bell' and 'chopper'). While, individual optimization of J_{div} alone picks 1 each from 8 classes, 2 from the class 'car' and none from the class 'bell'. It can be observed that exemplars chosen by J_{div} for classes 'glass', 'heart','flower' and 'apple' tend to be unusual members of the class. It also picks up the flipped car. Optimizing for both J_{div} and J_{rep} using Algorithm 5.3 picks one 'representative' exemplar from each class as shown in figure 5.4(c).

These exemplars picked by the three algorithms can be further used to label data points. Table 5.3 shows the confusion table thus obtained. For each data point, we find the nearest exemplar, and label the data point with the ground-truth label of this exemplar. For example, consider the row labeled as 'bell'. All the data points of the class 'bell' were labeled as 'pocket' by J_{rep} while J_{div} labeled 7 data points from this class as 'chopper' and 3 as 'pocket'. This confusion is largely due to both J_{rep} and J_{div} having missed out picking exemplars from this class. The proposed approach correctly labels all data points as it picks exemplars from every class.

KTH human action dataset: The next experiment was conducted on the KTH human action dataset [4]. This dataset consists of videos with 6 actions conducted by 25 persons in 4 different scenarios. For our experiment, we created a smaller dataset of 30 videos with the first 5 human subjects conducting 6 actions in the $s4$ (indoor) scenario. Figure 5.5(a) shows sample frames from each video. This dataset mainly consists of videos captured under constrained settings. This makes it difficult to identify the 'usual' or 'unusual' members of a class. To better understand the performance of the three algorithms, we synthetically added occlusion to the last video of each class. These occluded videos serve as the 'unusual' members.

	Glass	Heart	Apple	Bell	Baby	Chopper	Flower	Car	Pocket	Teddy
Glass	(10,10,10)									
Heart		(10,10,10)								
Apple		(0,1,0)	(8,7,10)			(2,0,0)			(0,2,0)	
Bell				(0,0,10)		(0,7,0)			(10,3,0)	
Baby					(10,10,10)					
Chopper		(2,0,0)	(8,0,0)			(0,10,10)				
Flower							(10,10,10)			
Car								(10,10,10)		
Pocket									(10,10,10)	
Teddy										(10,10,10

Table 5.3: Confusion Table. Entries correspond to the tuple $(J_{rep}, J_{div}, Proposed)$. Row labels correspond to the ground truth labels of the shape and the column labels correspond to the label of the nearest exemplar. Only non-zero entries have been shown in the table.

(a) Dataset

(b) J_{rep} (c) J_{div} (d) Proposed

Figure 5.5: (a) Sample frames from KTH action dataset [4]. From top to bottom action classes are { box, run, walk, hand-clap, hand-wave and jog }. 5 exemplars selected by: (b)J_{rep}, (c) J_{div} and (d) Proposed. Exemplars picked by J_{rep} correspond to { box, run, run, hand-clap, hand-wave } actions. While J_{div} selects { box, walk, hand-clap, hand-wave and jog }. Proposed approach picks { box, run, walk, hand-clap and hand-wave }.

Histogram of Oriented Optical Flow (HOOF) [146] was extracted from each frame to obtain a normalized time-series for the videos. A Linear Dynamical System (LDS) is then estimated from this time-series using the approach in [99]. This model is described by the state transition equation:

$$x(t+1) = Ax(t) + w(t)$$

and the observation equation

$$z(t) = Cx(t) + v(t)$$

where $x \in \mathbb{R}^d$ is the hidden state vector, $z \in \mathbb{R}^p$ is the observation vector, $w(t) \sim N(0, \Theta)$ and $v(t) \sim N(0, \Xi)$ are the noise components. Here, A is the state-transition matrix and C is the observation matrix. The expected observation sequence of model (A, C) lies in the column space of the infinite extended 'observability' matrix which is commonly approximated by a finite matrix

$$O_m^T = [C^T, (CA)^T, (CA^2)^T, \ldots, (CA^{m-1})^T]$$

. The column space of this matrix $O_m^T \in \mathbb{R}^{mp \times d}$ is a d-dimensional subspace and hence lies on the Grassmann manifold.

In this experiment, we consider the scenario when the number of classes in a dataset is unknown. We asked the algorithm to pick 5 exemplars when the actual number of classes in the dataset is 6. Figure 5.5(b) shows one frame from each of the videos selected when J_{rep} was optimized alone. It picks 1 exemplar each from 3 classes ('box','hand-clap' and 'hand-wave'), 2 from the class 'run' while misses out on 'walk' and 'jog'. On the other hand, J_{div} (when optimized alone) picks 1

each from 5 different classes and misses the class 'run'. It can be seen that J_{div} picks 2 exemplars that are 'unusual' members (occluded videos) of their respective class. The proposed approach picks 1 representative exemplar from 5 classes and none from the class 'jog'. The proposed approach achieves both a diverse selection of exemplars, and also avoids picking outlying exemplars.

Effect of Parameters and Initialization: In our experiments, the effect of tolerance steps (δ) for smaller values (< 0.1) has very minimal effect. After a few attempts, we fixed this value to 0.05 for all our experiments. In the first iteration, we start with $tol = 1$. With this value, Algorithm 5.2 accepts any swap that increases J_{div}. This makes output of Algorithm 5.2 after first iteration almost insensitive to initialization. While, in the later iterations, swaps are accepted only if they increase the value of J_{div} significantly and hence input to Algorithm 5.2 becomes more important with the increase in tol.

5.7 Conclusion and Future Work

In this chapter, we addressed the problem of selecting K exemplars from a dataset when the dataset has an underlying manifold structure to it. We utilized the geometric structure of the manifold to formulate the notion of picking exemplars which minimize the representational error while maximizing the diversity of exemplars. An iterative alternation optimization technique based on annealing has been proposed. We discussed its convergence and complexity and showed its extension to data manifolds and Euclidean spaces. We showed summarization experiments

with real shape and human actions dataset. Future work includes formulating subset selection for infinite dimensional manifolds and efficient approximations for this case. Also, several special cases of the proposed approach point to new directions of research such as the cases of vector spaces and data manifolds.

Chapter 6

Fast Nearest Neighbor on non-Euclidean Manifolds

6.1 Introduction

Large databases of images and videos are increasingly becoming common-place due to the growth of personal collections and Internet archives. In order to make these datasets more easily accessible to users, it is important to develop methods that allow fast retrieval and access to information. In many applications such as content-based image retrieval, texture classification, biometrics, and video mining, the problem is frequently reduced to searching for exemplars similar to a query in a large database. This is more formally known as similarity search or the nearest-neighbor (NN) problem. The problem of NN search has been studied for many years in the database and algorithms communities involving searching in a n-dimensional Euclidean space.

In the computer vision literature, over the past several years there has been significant research about what are good features and models for representing images and videos. Images and videos are usually represented by a concise set of features or models – such as shape [112], color/intensity histograms [147], SIFT [148], histogram of gradients [149], linear dynamic systems [99], covariance matrices [100], etc. As discussed, in detail in section 5.2, many of these features and models, however do

not lie in the Euclidean space. What this means is that the underlying distance function on the space is not the usual ℓ_2/ℓ_p norm but a highly non-linear function which is also in most cases computationally hard to compute.

Over the years, many advances have been made to understand the geometric properties of these varied spaces, and they have been utilized to devise more accurate inference and classification algorithms [114, 115, 125, 137]. Understanding the geometry allows one to define distances, leading to geodesics, etc. on these manifolds. In this chapter, we study the problem of fast NN search for data which lies in non-Euclidean manifolds. We provide a formal treatment of this problem for both exact and approximate NN search.

We start with devising techniques for exact but fast NN retrieval. This is achieved by creating a tree-based structure on the manifold. The tree structure is obtained by developing a hierarchical extension of the subset selection algorithm proposed in Chapter 5. As discussed there, this algorithm has significant advantages over clustering algorithms in handling imbalanced clusters. As is well-known, exact retrieval algorithm, although, much faster than exhaustive search still require $O(\log n)$ time for every query. This is further aggravated by the training time. The significant load in indexing with exact methods arises from the complexity of computing the distance function, and the complexity of computing centroids. These issues are generally not a concern in Euclidean spaces, but they become significant sources of complexity in non-Euclidean manifolds.

But, if one would be willing to accept a small reduction in the accuracy for a large gain in speed, efficient approximate NN techniques can be developed on non-

Euclidean manifolds. Towards this, we present a geodesic hashing technique which employs intrinsic geodesic-based functions to hash the training dataset.

6.1.0.1 Contributions

In this chapter, we make the following contributions:

- We propose a hierarchical extension of the diverse sampling technique proposed in Chapter 5. This equips us with an efficient tree structure for exact but fast NN retrieval on non-Euclidean manifolds.
- We propose an approximate but fast NN retrieval technique called geodesic hashing. This intrinsic technique employs orientation of query w.r.t. a geodesic to hash the training dataset.
- Geodesic distance computations are costly on a non-Euclidean manifolds and hence number of data-points colliding with a query point should be minimized. We address this by proposing a technique which optimally picks a set of geodesics to define the hashing family.

6.2 Related Work

Nearest-Neighbor rule is a widely used technique in pattern recognition and computer vision. It is a lazy learning technique where all the training points are stored and during testing, distance of the query point from each training point is evaluated to identify the nearest training-point. This causes exhaustive NN search to

grow linearly with the number of training data points. This becomes significantly more computationally intense when distance computation between every pair of points is intensive. For instance, consider the Grassmann manifold ($\mathcal{G}_{m,p}$) which is the space of p-dimensional subspaces embedded in $\mathbb{R}^m$. Geodesic computation between a pair of points on $\mathcal{G}_{m,p}$ requires $O(p^3)$ time. This intensive nature of NN has led to interest in developing fast NN search techniques both for Euclidean and non-Euclidean spaces.

In the relatively better understood domain of searching in Euclidean spaces, numerous techniques that drastically reduce the search time (both algorithms and structures) have been developed over the last couple of decades. These techniques have been studied in many different fields and can be broadly categorized as (a) exact, and (b) approximate NN search techniques. [150, 151, 152] provide few recent surveys of these techniques. Another notable survey, especially from computer vision perspective, is by Kumar *et al.*[153] which provides a comprehensive comparison of exact search methods applied to patch-based image searching.

Exact methods usually rely on space partitioning. Examples of this approach include quad-trees, $k-d$ trees and ball-trees. The resulting data structure is represented as a tree with the root node being the entire dataset, child nodes representing partitions and leaf nodes representing individual data points. PCA trees and $k-d$ trees are "projective trees" as they split points based on their projection into some lower-dimensional space. These techniques require the coordinates of the points to split data hierarchically. This restricts the application of these techniques in metric spaces where only a metric is defined on the space. Another disadvantage with $k-d$

trees is that for dimensions larger than 10, the algorithm becomes very inefficient and tends towards a linear scan algorithm. PCA trees overcomes the restriction of $k-d$ tree by splitting data points such that the boundary is aligned to one of the axes. So, now it can find the variance of the data and split according to the hyperplane that splits the maximum variance.

Approximate methods in Euclidean spaces became popular when Locality Sensitive Hashing (LSH) algorithms [154, 155] were introduced. The original LSH algorithm was proposed by Gionis *et al.*[155] which was tuned for 'Hamming'-spaces i.e., spaces where the underlying distance function is the Hamming distance. This was extended to Euclidean spaces in [156]. A good introduction and survey of these methods can be found in [157]. The core idea of LSH techniques is to efficiently construct binary codes for high-dimensional points while ensuring that points that are nearby in high-dimensional space are close even in the binary code space. Brief overview of LSH technique is discussed later in section 6.5.1.

All these methods rely heavily on the assumption that points are in $\mathbb{R}^n$ with the underlying metric being the ℓ_2 norm. These methods can also be applied directly to non-Euclidean data-points if the manifold of interest can somehow be embedded into the Euclidean space. In this chapter, we discuss that it is important to consider the Riemannian geometry of manifolds to more systematically address this problem. This is because there exists several analytic manifolds which cannot in any easy way be embedded into an ambient Euclidean space. For instance, the space of linear subspaces or the Grassmann manifold is best treated as a quotient space of the orthogonal group and there is no easy natural embedding into an ambient vector

space [158].

Nearest Neighbor search in metric spaces when only an underlying metric is known has also been studied in literature. Several NN algorithms have been devised for indexing data with arbitrary distance functions such as vantage-point trees, metric trees and multi-vantage point trees. An excellent survey of these methods can be found in [150]. These are exact methods of searching in metric spaces. Unfortunately, tree-based indexing methods such as these often suffer from the curse of dimensionality. For large dimensions the performance is not significantly different than simple brute-force search. Approximate search methods in arbitrary metric spaces have also been proposed [159]. Indyk and Motwani [154] provided algorithms for approximate searches when the underlying distance function is the ℓ_∞ norm. In some cases the underlying non-Euclidean distance measure can be replaced with another which can be efficiently computed. Several methods have been proposed for embedding arbitrary spaces into Euclidean or pseudo-Euclidean space [160, 161]. Embedding methods substitute a fast approximate distance for the original distance hence are approximate search methods.

We note that fast NN retrieval for manifold data points is still an emerging area and hence the literature on it is scarce. The work of [162] shows how to adapt the standard k-d tree algorithm to low-dimensional manifolds whose structure is unknown. The work on approximate nearest subspace search [163] is an example of indexing on non-Euclidean manifolds. However, this work is limited in applicability to subspaces; moreover, it does not exploit the Riemannian geometric interpretation of subspaces.

Recently, there has been two works addressing this problem of adapting NN techniques from vector space to non-Euclidean manifolds. Turaga and Chellappa [1] proposed an approximate NN technique via embedding of data into the tangent space at a pole using the log-Euclidean mapping. Similarly, Chaudhry and Ivanov [164], used log-Euclidean embedding to develop a spectral hashing method. However, both these techniques are based on Euclidean embedding of the Riemannian manifold through tangent spaces.

6.3 Examples of computations on Non-linear Manifolds

In this section, we present an overview of various manifolds that frequently appear in vision literature and their associated non-linear distance functions. This has been presented in Table 6.1 which is reproduced from [1]. As can be seen, the distance computations (and centroid computations) can be of significant complexity for these manifolds. This necessitates the development of fast NN search techniques on non-Euclidean manifolds.

6.4 Exact Nearest Neighbor using Tree Structure

In this section, we discuss an exact NN search algorithm that uses a tree-based structure on manifolds. In Euclidean space, exact methods like quad-trees, and $k-d$ trees, usually rely on space partitioning. In this set of approaches, the resulting data structure is a tree with the root node being the entire dataset, child nodes representing partitions, and leaf nodes representing individual data points.

Manifold	Numerical Representation	Dimension	Commonly used Distance functions	Applications
Spherical manifold	n-vector with unit norm	$n-1$	$d(X_1, X_2) = \cos^{-1}(\lvert x_1^T x_2 \rvert)$	Kendall's shape space [112], probability density functions [165]
Stiefel manifold	$n \times k$ orthonormal matrix	$nk - \frac{k(k+1)}{2}$	No closed analytical form.	Face recognition [134], dimensionality reduction [135]
Grassmann manifold	$n \times k$ orthonormal matrix	$k(n-k)$	$d(X_1, X_2) = \sum_i \lVert\theta_i\rVert^2$, where $\cos(\theta_i)$ are singular values of $X_1^T X_2$	Image set modeling [129, 130, 131, 132], dynamic models [99, 137]
Covariance matrices	$n \times n$ symmetric pos. def. matrix	$\frac{n(n+1)}{2}$	$d(X_1, X_2) = \sqrt{\sum_{i=1}^{n} ln^2 \lambda_i(X_1, X_2)}$, where $\{\lambda_i\}$ are generalized eigenvalues of $\lambda X_1 v = X_2 v$	Region descriptors [100, 126, 125], diffusion tensor imaging [127]

Table 6.1: Examples of various manifolds that appear in vision applications. The table shows the highly non-linear and sometimes computationally intensive distance functions on the manifolds. Depending on data, the dimension of the manifold can be quite high. This table has been reproduced from [1].

Unfortunately, these methods use coordinates of the points and hence rely heavily on the assumption that points are in $\mathbb{R}^n$ with the underlying metric being the ℓ_2 norm. Alternatively, clustering and pivot-based tree structures instead rely on pairwise distances and are extensible to manifolds. Here, the general goal is to divide a set into subsets of elements close to each other in the same subset. The resultant tree structure is equivalent to a Voronoi partition of the data-points.

Recently, several clustering techniques like k-means [114], Mean-shift [122], etc. have been extended to manifolds. These techniques, in particular, their hierarchical version [166], give a tree-structure for fast NN search. These techniques have been known to work well with balanced classes but in the case of unbalanced classes, they get skewed by the dominant class. Unbalanced classes are commonplace in pattern recognition and computer vision. For instance, in many classification problems, the positive class is much less frequent than the negative class e.g., object detection, credit card fraud detection, etc. In order to overcome this, in our earlier works [104, 167], we emphasized the need for having diverse exemplars/clusters/pivots and proposed an algorithm to achieve clustering on the manifold. We extend this algorithm to hierarchically create tree structure on the manifold.

Similar to other tree construction algorithms, the proposed algorithm begins by assigning all data-points to the root node, and then recursively partitions points into one of several children of the node. Every node of the tree is considered as a cluster and is represented by a data-point and then the query is compared with this representatives. We start from the root node and pick k diverse exemplars using the diverse column subset selection (CSS) algorithm proposed in [167]. Every data

point in the root node is then assigned to the closest exemplar. The natural choice of representative data-point for each cluster would be the mean or the data-point closest to the mean. Each cluster is then recursively divided into k clusters. This recursion terminates when the cluster at the leaf node has fewer than M data-points.

Query proceeds in a top-down fashion. Query data is first compared to the k data-points at the first depth. The pivot closest to the query point is selected and then the query is compared with k data-points at second depth which are the children nodes of the selected data-point. Several pruning rules mostly based on triangle inequality have been developed for speeding up the NN search by avoiding the traversal of some parts of the tree. In this chapter, we do not consider those techniques. We adopt the triangular-inequality based pruning rules suggested by Fukunaga and Narendra [168]. Similar to other tree-based algorithms, the complexity of this algorithm is $O(\log n)$, where, n is the number of data-points.

6.5 Intrinsic Approximate Nearest Neighbor Search

In this section, we introduce Geodesic Hashing, an intrinsic approach for approximate NN search on Manifolds. But before that, we review the hashing technique and the property of Locality Sensitive Hashing (LSH) method.

6.5.1 Hashing

Hashing was originally studied in the field of cryptography and text databases which involved entities such as passwords, names, addresses, etc. where finding

exact matches was the key requirement. Hashing image and video data brought additional challenges since no two semantically related images or videos are exactly the same. This brought about a new class of techniques - LSH [155] - that could answer approximate NN queries in a fast manner.

A good introduction and survey of LSH can be found in [157]. Here, we briefly review the basic concepts of Euclidean LSH. LSH attempts to solve a problem called the (r, ϵ)-Neighbor problem. The problem is described as follows: Given a database of points $X = \{x_i\}$ in $\mathbb{R}^n$, and a query q, if there exists a point $x \in X$ such that $d(x, q) \leq r$, then with high-probability, a point $x' \in X$ is retrieved such that $d(x', q) \leq (1 + \epsilon)r$. LSH solves this problem by constructing a family of hash-functions $\mathcal{H}$ over X called (r_1, r_2, p_1, p_2) locality-sensitive, if for any $u, v \in X$

$$d(u, v) \leq r_1 \quad \Rightarrow \quad Pr_{h \in \mathcal{H}}(h(u) = h(v)) \geq p_1 \tag{6.1}$$

$$d(u, v) \geq r_2 \quad \Rightarrow \quad Pr_{h \in \mathcal{H}}(h(u) = h(v)) \leq p_2 \tag{6.2}$$

A (r_1, r_2, p_1, p_2) locality-sensitive family of hashing function solves the (r, ϵ)-Neighbor problem by choosing $r_1 = r$, and $r_2 = (1+\epsilon)r$. Popular choices of h include random projections i.e., $h(v) = sgn(v^T r)$ where r is a randomly chosen unit-vector, and sgn is the signum function. In this case, h is binary valued taking values in $\{+1, -1\}$. A generalization of this is termed random projections using 'p-stable' distributions [156], where

$$h(v) = \left\lfloor \frac{v^T r + b}{w} \right\rfloor$$

where r is a randomly chosen direction whose entries are chosen independently from a stable distribution, and b is a random number chosen between $[0, w]$. In this case,

the hash function takes on integer values. A $k-$bit hash is constructed by appending k randomly chosen hash-functions $H(x) = [h_1(x), h_2(x), \ldots, h_k(x)]$. Thus, $H \in \mathcal{H}^k$. Then, L hash tables are constructed by randomly choosing $H_1, H_2 \ldots H_L \in \mathcal{H}^k$. All the training examples are hashed into the L hash tables. For a query point q, an exhaustive search is carried out among the examples in the union of the L hash-buckets indexed by q. Appropriate choices of k and L ensure that the algorithm succeeds in finding a (r, ϵ)-NN of the query q with a high probability.

6.5.2 Geodesic Hashing (GH)

We now discuss Geodesic Hashing (GH) which utilizes geodesics on the manifold to hash data lying on a non-Euclidean manifold. Let $\mathcal{M}$ represent the manifold, and $\mathfrak{X} = \{x_1, x_2, \ldots x_n\} \in \mathcal{M}$ be the database of n points. The first step in our formulation is to define a family $\mathcal{H}$ of hash-functions over $\mathcal{M}$ which is (r_1, r_2, p_1, p_2)-sensitive. We obtain this, by seeking the generalization of the binary-valued random projection ($h(v) = sgn(v^T r)$) hashing function to the Riemannian manifold.

In Euclidean space, this family of hash-function is created by randomly picking a projection direction r. Binary hash value for a data-point v is then obtained by looking at the orientation of v w.r.t. r. In Euclidean geometry, the notion of 'origin' is very well-defined. Hence, hashing family is defined well by selecting a set of directions. On the other hand, non-Euclidean manifolds do not have a point that naturally serves as the 'origin' for the manifold. One could attempt to define an analog of the origin for manifolds by computing the intrinsic mean of

the data and use it as a 'pole' to project data-points on the tangent space of the pole. This approach was recently taken by [1]. As the tangent space is Euclidean, LSH techniques can directly be applied on the projected data. This approximation works reasonably well if the data-spread is small. But, as the data-spread increases, the approximation increasingly becomes crude. In order to overcome this, Chaudhry and Ivanov [164], proposed computing poles for individual clusters. Individual data-points are then hashed in the tangent space of the closest pole. This reduces the error in data-approximation. However, this method relies heavily on finding the nearest pole and incurs large errors for points near the boundary of two clusters. Moreover, choice of number of clusters could be critical to the performance of the hashing technique.

Attempting to define one or few poles which serve as the analog of 'origin' for non-Euclidean manifolds followed by Euclidean embedding of data, leads to crude approximation of data. We can better generalize hashing for manifolds by looking at the configuration of data points w.r.t. a randomly chosen geodesic on the manifold. Geodesics on manifolds are analogs of 'lines' in Euclidean spaces and computing a query point's orientation w.r.t. the geodesic does not require us to make any approximation. Consequently, this helps us to get rid of the reliance of LSH techniques on the artificially created notion of origin. This, in turn, leads to a better way of hashing antipodal points.

In order to formulate this, let $g_{a,b}$ represent a geodesic specified by the pair of points (x_a, x_b) s.t. both $x_a, x_b \in \mathcal{M}$. Now, given a query point, $q \in \mathcal{M}$, one can look at the projection of q on $g_{a,b}$. In Euclidean space, projection of a point on a vector

is a relatively cheaper operation. But, in Riemannian manifolds, projecting a query point on a geodesic, is a costly optimization problem [114]. Hence, choosing any projection based functions would significantly increase the query time. So, instead we can look at the orientation of the point q w.r.t. the geodesic $g_{a,b}$. Specifically, this is obtained by defining a binary function $h_{(x_a,x_b)}(q) : \mathcal{M} \rightarrow \{-1, +1\}$ such that,

$$h_{(x_a,x_b)}(q) = sgn(Log_{x_a}(x_b)^T Log_{x_a}(q)) \tag{6.3}$$

So, given a set of geodesics $\mathbb{G}$, this provides us a family of hashing functions

$$\mathcal{H}_{GH}(\mathbb{G}) = \{h_{(x_a,x_b)} \quad | \quad g_{a,b} \in \mathbb{G}\}$$

It can be noted here that the family of functions defined using equation (6.3) is a very rich family and is specified completely by $\mathbb{G}$. Having defined the family of hash-functions, we can append k-random binary hash functions h sampled from $\mathcal{H}_{GH}(\mathbb{G})$ to define a k-bit hash function. Then, L hash tables are constructed by randomly choosing $H_1, H_2, \ldots, H_L \in \mathcal{H}^k$. This way, we obtain an intrinsic indexing and retrieval method on the manifold in the LSH framework.

Now, in order to verify the above mentioned intuition, we synthetically generated a dataset on $\mathcal{G}_{9,3}$. We adopted the procedure described in [169] to generate this dataset which has 4 randomly generated clusters with 200 points each. We empirically computed the probability of collision between two points (u, v)

$$Pr_{h\in\mathcal{H}}\big(h(u) = h(v)\big)$$

Figure 6.1 shows the probability of collision with increase in the geodesic distance between the points averaged over 5 such randomly generated datasets. We computed

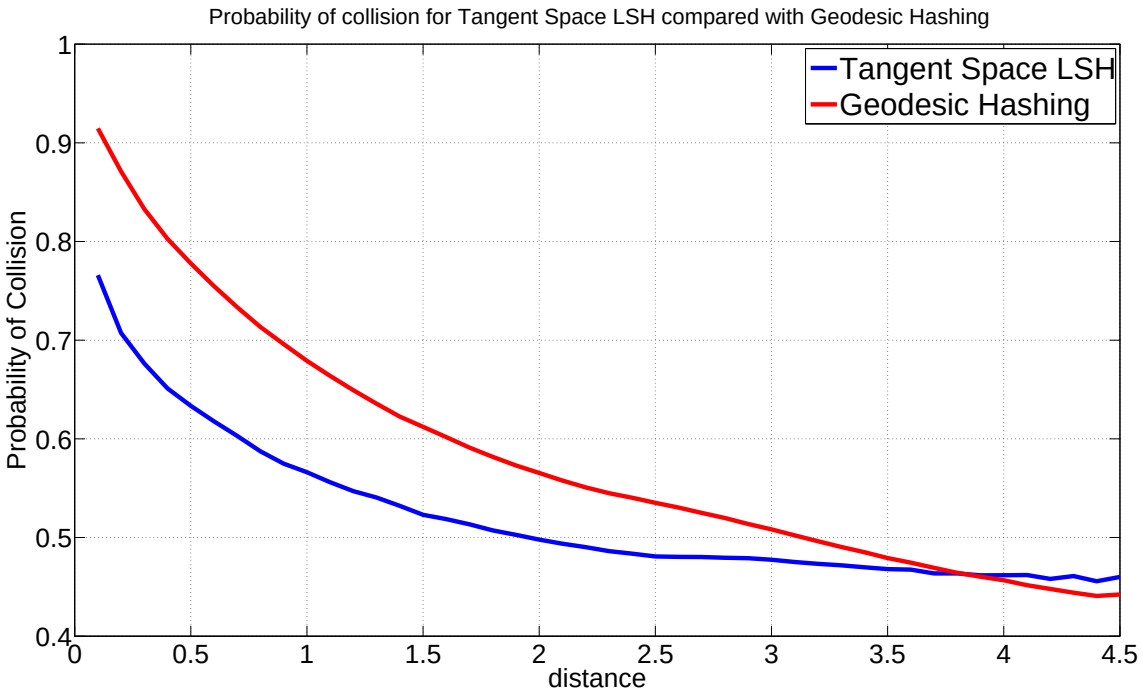


Figure 6.1: Comparison of Probability of collision for Tangent space LSH [1] with that for Geodesic Hashing.

this for both tangent space LSH [1] (blue curve) and Geodesic Hashing (red curve). It can be noted that tangent space LSH due to approximate embedding of data sees a deterioration in the collision probability even for points that are nearby.

6.5.2.1 Satisfying LSH property

In the previous section, given a set of geodesics $\mathbb{G}$ from the manifold $\mathcal{M}$, we created a family $\mathcal{H}_{GH}(\mathbb{G})$ of hashing functions. Now, we verify if the proposed family satisfies the LSH properties which requires that $\mathcal{H}_{GH}(\mathbb{G})$ should satisfy equations (6.1-6.2) over $\mathcal{M}$. For this, similar to previous section, we generated a synthetic dataset on $\mathcal{G}_{9,3}$ and empirically evaluated the probability of collision for a randomly chosen set of geodesics $\mathbb{G}$. The collision probability, thus computed, has been aver-

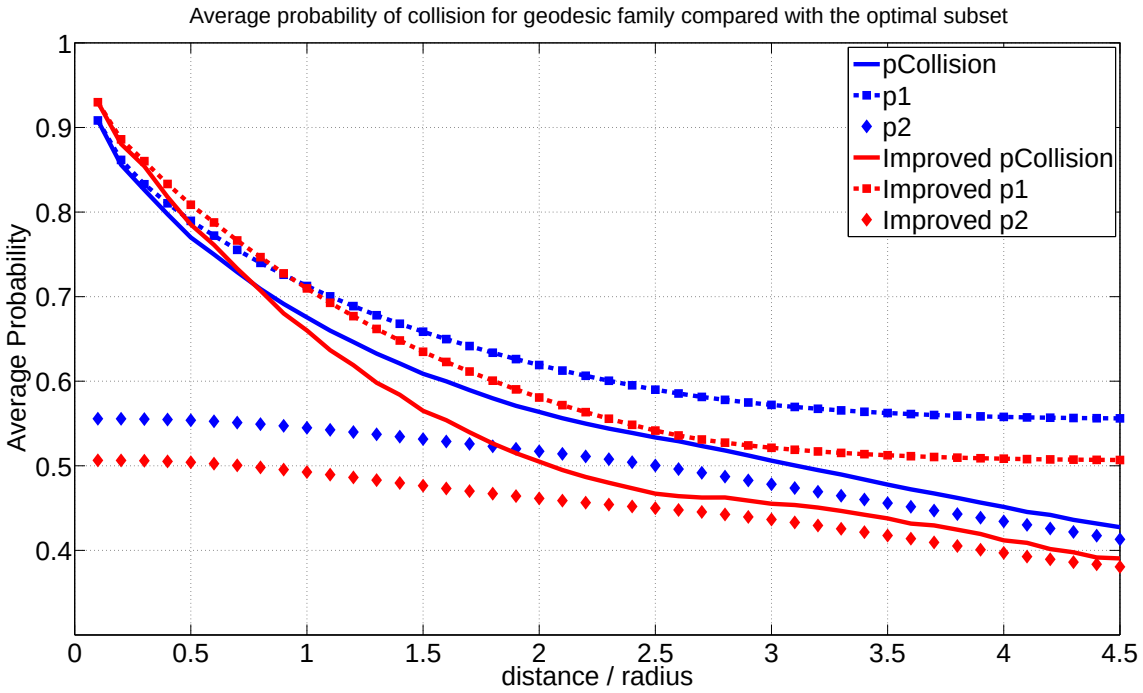


Figure 6.2: LSH property evaluation. Curves in blue show the empirical evaluation of LSH properties for Geodesic Hashing. The blue curve marked as 'pCollision' shows the monotonic decrease in probability of collision with increase in distance. Similarly, p_1 and p_2 values, as defined in equations (6.1-6.2), are shown with radius values (r_1, r_2) respectively on the x-axis. The red curves show the improvement in collision probabilities along with improvements in p_1 and p_2 by optimally selecting the set of geodesics.

aged over 20 random selection of $\mathbb{G}$ and is shown in Figure 6.2. This figure shows that the probability of collision monotonically decreases with increase in pairwise distance thus satisfying an important LSH property.

Moreover, as defined in equation (6.1), we computed the probability of collision p_1 for points within a ball of radius r_1. Figure 6.2 shows the variation of p_1 with the increasing radius r_1. Similarly, this figure also shows the variation of p_2 with r_2 defined in equation (6.2). It can be seen here that for any choice of (r_1, r_2) such that $r_2 > r_1$, corresponding collision probabilities satisfy $p_1 > p_2$ and $p_1 > 0.5$, hence guaranteeing the usefulness of this LSH family [155].

6.5.3 Optimal set of geodesics

When compared to the exact NN techniques, LSH techniques require us to compute distance of a query point with a very small number of data-points. Hence, in Euclidean space, where the pairwise distance computation is relatively cheaper, it suffices to have defined a family of hashing functions. However, as discussed earlier and shown in Table 6.1, in the case of non-Euclidean manifolds, geodesic distance computations are costly operations. This implies that a further reduction in the average number of points that collide with a query point for a hashing family $\mathcal{H}_{GH}(\mathbb{G})$ is required. This can be achieved by optimally selecting the set of geodesics $\mathbb{G}$, with an eventual goal to significantly reduce the query time. Although, optimal selection of $\mathbb{G}$ requires more processing during training, it can significantly reduce the query time.

This requires us to address various criteria that the set of geodesics should meet w.r.t. the statistics of the database $\mathfrak{X}$, in order to allow fewer collisions while retaining high accuracy. On an individual level, a good geodesic should satisfy the following criteria:

1. The binary sequence generated by a geodesic for $\mathfrak{X}$ should have equal probability of ones and zeros.

2. Geodesics should have large value of p_1 and smaller value of p_2.

Moreover, the group of geodesics $\mathbb{G}$ should satisfy an additional criteria that the binary sequence generated by a geodesic should be independent of that generated by another geodesic. Weiss *et al.*[170] presented a similar discussion from the perspective of a good code. Formally, these criteria can be written as the following: Let us denote the probability of collision of nearby points by,

$$C_r(q,p) = Pr_{h\in\mathcal{H}_{GH}(\mathbb{G})}\left[h(q) = h(p)\right] \qquad \text{when} \qquad d_g(q,p) \leq r$$

Then, the optimization problem can be written as:

$$\max_{\mathbb{G}} \;\; C_r(q,p)$$

s.t. if $g_{a,b} \in \mathbb{G}$, then

$$Pr_{q\in\mathfrak{X}}[h_{(x_a,x_b)}(q) = 1] = 0.5$$

and for $g_{a',b'} \in \mathbb{G}$

$$Pr_{q\in\mathfrak{X}}[h_{(x_a,x_b)}(q) = h_{(x_{a'},x_{b'})}(q)] = 0.5$$

6.5.3.1 Optimization

An optimal selection of $\mathbb{G}$, with $|\mathbb{G}| = \tau \geq kL$, that satisfies the above criteria can be achieved by the following procedure: The set of geodesics on the manifold $\mathcal{M}$ is very large. So, in order to reduce the search space, we restrict ourselves to the geodesics formed by randomly selecting a pair of points from $\mathfrak{X}$. Let $\mathbb{G}_{\mathfrak{X}}$ represent this set of geodesics. It should be noted here that even after this reduction in search space, the number of possible geodesics $|\mathbb{G}_{\mathfrak{X}}| = \binom{n}{2}$ is still very large. So, we randomly select a set of geodesics $\mathbb{G}_R \subseteq \mathbb{G}_{\mathfrak{X}}$ with $|\mathbb{G}_R| = m >> \tau$. This is followed by evaluating criterion 1 for individual geodesics over a smaller uniformly sampled subset $\mathcal{X} \subseteq \mathfrak{X}$. This provides us $\mathbb{G}_E \subseteq \mathbb{G}_R$, the set of geodesics with equal probabilities of ones and zeros. For the set $\mathbb{G}_E$, p_1 on $\mathcal{X}$ is computed. We then greedily populate the set $\mathbb{G}_O$ with geodesics of high p_1 values on the condition that they are not correlated to any geodesic $g_{a,b}$ already in $\mathbb{G}_O$.

Figure 6.2, shows the improvement in the probability of collision w.r.t. distance, and p_1 w.r.t. r_1, and p_2 w.r.t. r_2. As expected, the optimized family of geodesics increases the collision probability for nearby points and reduces it for far away points. In other words, points close to a query point will have better probability of colliding and points far away will have lesser probability of colliding. This signifies that for the same accuracy, the number of colliding points is reduced.

6.5.3.2 Variation with parameters

We now study the variation of performance with the parameters k (number of bits) and L (number of buckets). We performed this analysis on the synthetic dataset described earlier in the previous section. In this case, 200 points for training and 75 points for testing were generated in each class. Figure 6.3(a) shows the classification accuracy variation, while Figure 6.3(b) shows the average accuracy in finding the true nearest neighbor. It can be seen here that for errors in both classification and finding true NN decreases with the number of tables. This is quite expected and is explained by the fact that for the same number of bits, the number of training points colliding with query point increases with the number of tables. This improves the probability of finding the true NN at the cost of distance computations with larger number of training points as shown in Figure 6.3(c). On the other hand, more training points collide with the query point with decreasing number of bits, thus increasing the number of geodesic computations required and reducing the error in searching for true NN.

6.6 Conclusion

In this chapter, we first developed the hierarchical extension of diverse clustering technique proposed earlier in this dissertation. This hierarchical extension has been used to create a tree structure on the manifold for exact but fast NN. We then proposed an intrinsic geodesic hashing technique for the problem of approximate NN search on non-Euclidean manifolds. This technique utilizes the orientation of

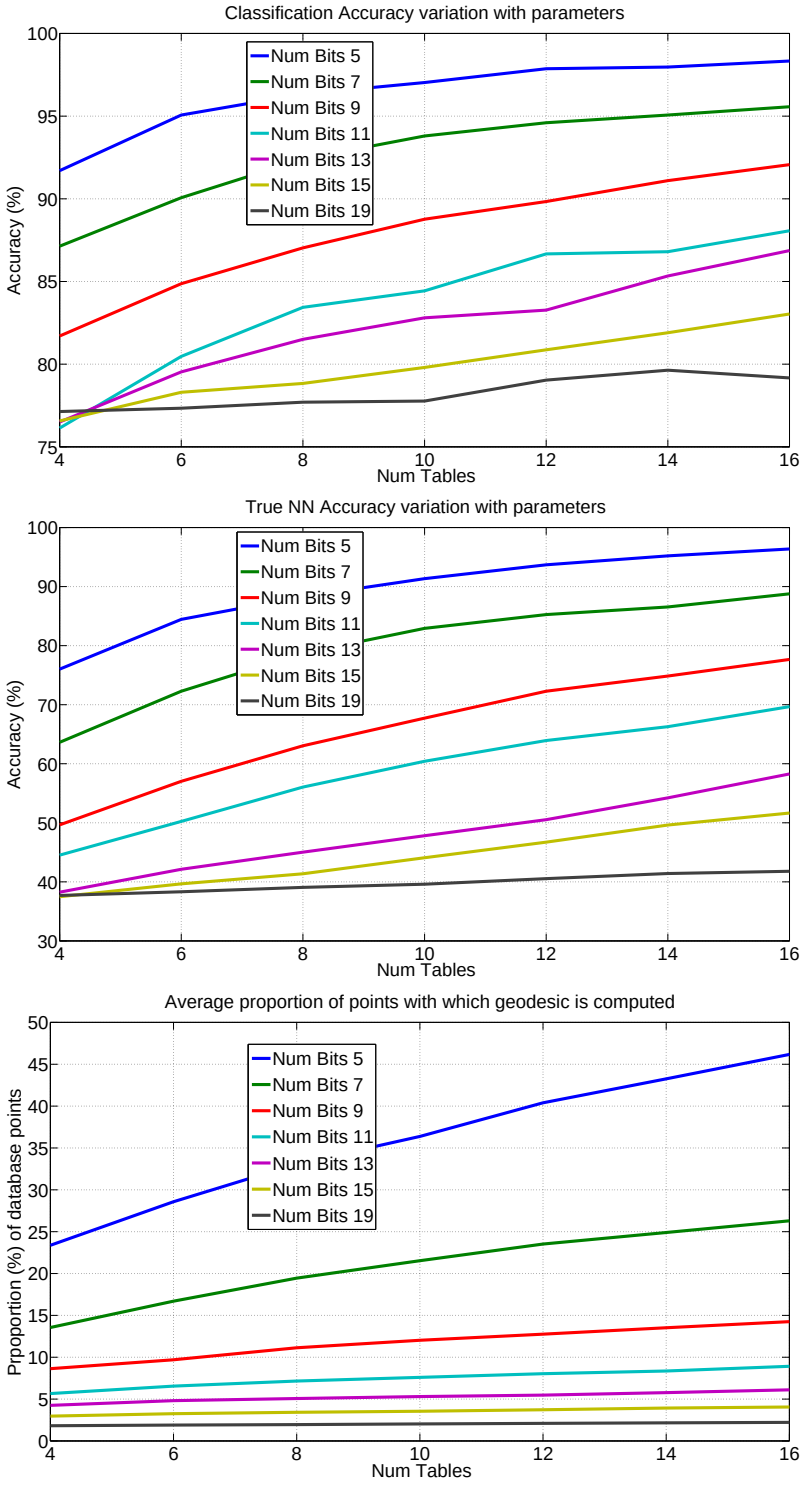


Figure 6.3: Geodesic Hashing (a) Classification accuracy, (b) True NN accuracy, and (c) Number of Geodesic Computations

a data-point w.r.t. a set of geodesics to hash the data. A technique is devised to optimally select the set of geodesics, which in turn defines the hashing family. In this ongoing work, we have shown encouraging preliminary results on synthetically generated data lying on Grassmann manifold. As part of this work, we intend to show results on several real datasets.

Chapter 7

Moving Vistas: Exploiting Motion for Describing Scenes

7.1 Introduction

Scene recognition is an area of active research in computer vision where the goal is to identify an image as belonging to one of several classes such as mountains, beaches, or indoor-office, etc. Recognition of scenes by humans is usually explained in one of two ways. The first approach suggests scene recognition proceeds in a hierarchical manner by processing and understanding objects and their contextual relationships [171]. This line of thought has been used extensively in computer vision applications such as in [172, 173, 174]. Alternately, in many cases humans also recognize scenes by holistically understanding the properties of a scene [175] instead of fine object level descriptions. This line of thought has also resulted in several features and algorithms for scene analysis [176, 177].

While there exists a rich literature for modeling static scenes, much less attention has been devoted to studying if and how motion in the scene can be exploited for describing scenes. Motion information is useful both at object-level and at global-level. In this chapter, we investigate how the motion of scene elements themselves can provide more descriptive representations of videos than simply using their static appearances.

Figure 7.1: Consider the 2 frames on the left of the arrows. Both suggest a snow-clad mountain scene. But when temporal evolution of the scene is considered, as shown on the right, further information about 'avalanche' in the bottom video is revealed.

For instance, consider the 2 frames on the left of the arrow in Figure 7.1. Both the frames suggest a snow clad mountain scene. But the sequence of frames (to the right of the arrows) gives further information that the bottom video is an avalanche scene. Here, the motion of the scene-elements (in this case, the snow) provides better fine-grained description of the scene. Similar examples can be imagined such as differentiating haphazard traffic from smooth traffic, a sea wave from a whirlpool, and numerous other examples. As these examples illustrate, a scene may be composed of the same elements in similar relative spatial positions, but the manner in which they move in relation to each other can be exploited to obtain a better, fine-grained distinction in scene categories.

Purely from a recognition perspective, a number of experimental studies have shown that motion information helps by enhancing the recovery of information about shape [178], edges [15], views [16] etc. In this chapter, we propose to use scene-motion to *enhance the description of a scene*, and not merely as a method to improve

static scene recognition. We show via examples and experiments in this chapter that motion indeed provides informative cues about the scene. We demonstrate these ideas via recognition experiments, and labeling of videos with semantically meaningful attributes.

7.2 Related Work

There has been significant interest in recent years to recognize scenes. Towards this direction, several approaches have emphasized utilizing the presence of objects to recognize the scene. The co-occurrence of objects and actions with scene classes [173, 179] has been utilized to recognize the scenes. In a similar approach, relationships among locations of observed objects are modeled by prior distributions [172].

A parallel theme has been to use global cues like color histogram, power spectrum, global frequency with local spatial constraints [177, 180, 181, 176]. Local features have also been used to build a global descriptor of the image [182, 183]. The key idea in using these global cues is that a holistic approach, without analysis of its constituent objects, provides effective cues about its semantic category. Oliva and Torralba further incorporated the idea of using attribute-based descriptions of scenes [176]. Recently, both local and global discriminative information was fused together [184] to recognize indoor scenes.

These approaches have focused mainly on analyzing scenes from a static image. But none of these approaches have exploited motion for analyzing these scenes.

Motion information, as we discussed, provides crucial cues about the dynamics of the scenes. Several methods to model dynamics have been considered in vision literature - some well-known [185, 97], some not so well-known [186]. Here, we briefly review them.

Doretto *et al.* [185] proposed linear dynamical systems (LDS) as models for dynamic textures. The LDS model is shown to be well-suited to capture second-order stationary stochastic processes and an effective generative model. This basic LDS model and its proposed extensions [187, 188] have been used for segmentation and recognition of dynamic textures and human activity recognition [136]. However, the first-order Markov property and linearity assumption in the model make it restrictive for modeling unconstrained dynamic scenes. We will show in our experiments that this is indeed the case.

The patch based bag-of-words approach has been used to model objects [189], scenes [182, 183], and human actions [3, 97] from videos. This approach uses local patches to build global representations. Due to its local nature, it can robustly handle dynamic backgrounds and moving cameras. When applied to scenes, this formalism does not distinguish scenes that locally may look similar, but globally appear to be different, e.g. consider avalanche vs. iceberg collapse.

7.2.1 Overview of our Approach

In this work, we focus on ways to model the dynamics of the scenes and how this information may be utilized to describe scenes better. Some challenges that

make application of standard methods difficult are: a) Scenes are unconstrained and 'in-the-wild', and b) the underlying physics of motion is either too complicated or very little is understood of them. Thus, this renders many of the standard methods of dynamic modeling unsuitable for this task. Consider, for example the last row in Figure 7.2, which shows 6 different examples of videos from the same class 'Tornado'. The large variations in appearance due to scale, view, illumination, and background can be easily perceived.

Motivated by these challenges, we model dynamic scenes using a chaos-theoretic framework. Chaos theory is concerned with understanding dynamical systems whose structure is unknown. Further, in such systems small changes in initial conditions result in huge variations in the output. But, the underlying process is not entirely random – infact, there exists a deterministic component which cannot be characterized in a closed form. These properties make this framework attractive in the study of in-the-wild dynamic scenes. This framework has been applied to gait modeling by Perc [186] and human action recognition by Ali *et al.* [190]. Apart from these applications, this framework has not been widely adopted in computer vision literature. This is probably because most problems in computer vision involving motion such as structure-from-motion, human actions, dynamic textures are usually constrained in nature, for which simple models such as second-order Gauss-Markov, or a constant-velocity model suffice.

7.3 Contributions

Our specific contributions are:

- We study the role of dynamics of scene elements for improved representation of scenes.
- Then, we address the problem of finding accurate and generalizable ways for characterizing the hard-to-model dynamics of unconstrained scenes via chaotic systems. We show how these dynamic features lead to better recognition of videos using a dataset of 'in-the-wild' dynamic scenes.
- We introduce dynamic attributes and show semantic organization of videos using these attributes.

Outline: Section 7.4 discusses the role that motion information and its attributes can play in scene representation. In section 7.5, we propose a framework based on chaotic systems to characterize unconstrained scene dynamics. In section 7.6, we discuss the dynamic scene dataset and provide extensive experimental results. Finally, in section 7.7, we conclude and discuss the benefits and limitations.

7.4 Motion Attributes

Without taking recourse to object-level descriptions, motion information of a scene can be described from a global perspective by attributes such as

Degree of Busyness: Dynamic scenes can be characterized by the amount of activity happening in the video. Consider scenes such as sea-waves or a traffic

scene. These scenes are characterized by a high degree of detailed motion patterns. On the other hand, scenes such as a waterfall appear largely unchanging and the motion typically occurs in a small portion of the scene.

Degree of Flow Granularity: Scenes that contain motion can also be distinguished based on the granularity of the scene's structural elements that undergo motion. Structural elements such as the falling rocks in a landslide are coarse in granularity, whereas the waves in an ocean are of a fine granularity.

Degree of Regularity: This attribute characterizes the quality of motion in terms of its regularity or irregularity. Even though traffic scenes may consist of the same level of busyness and granularity, chaotic traffic exhibits a high degree of irregular or random motion, whereas smooth traffic scenes exhibit a regular motion pattern.

As these examples illustrate, motion information can be exploited significantly to augment the spatial descriptions of scenes. In section 7.6.3, we will show how these motion-attributes can be learnt using dynamic features to provide semantically meaningful organization of videos.

7.5 Modeling Dynamic Scenes

Most models describing the dynamics of a system, assume that there exists an underlying mapping function f that synthesizes the next observation from the current observation, i.e. $y(t) = f(y(t-1)) + n(t)$. The structure of the function f encodes the dynamics of the underlying system. By making assumptions on the

structure such as in the case of LDS [185], model fitting can be efficiently solved. However, in our application, the 'wild' and uncontrolled setting does not lend itself to making simplifying assumptions about the system. However, even without making any specific assumptions on the function f, using the theory of chaotic systems, it is possible to derive certain 'invariants' of f purely from the sequence of observations $y(t)$. We next describe the chaotic system framework that provides a solution to this problem.

7.5.1 Chaotic Invariants

Fundamental to chaos theory is the fact that all variables in a deterministic dynamical system influence one another and thus are generically connected. Thus, every subsequent point of a given measurement is the result of an entangled combination of influences from all other system variables [186]. Due to space constraints, we briefly mention the steps below and refer the reader to [186, 190] for further details.

Consider a one-dimensional time series $\{x_t\}_{t=1}^{n}$. The first step is the reconstruction of attractor dynamics from the given time series. An *attractor* of a dynamical system is defined as the region of phase space that over the course of time (iterations), attract all trajectories emanating from some range of starting conditions [191]. The reconstruction is achieved by constructing the set of vectors that result from the concatenation of m time delayed version of x_t i.e., the vector $[x_1, x_{\tau+1}, ..., x_{(m-1)\tau+1}]$. These vectors are called the state-variables. The space of

all state-variables is called the phase space. Here, m is the embedding dimension and τ is the embedding delay.

The embedding delay τ is calculated from the given time series using the mutual information [192]. For this, the range of the given values $[min(x_t), max(x_t)]$ is first divided into equal bins.

$$I(\tau) = -\sum_{s=1}^{b}\sum_{q=1}^{b} P_{s,q}(\tau) log \frac{P_{s,q}(\tau)}{P_s(\tau)P_q(\tau)} \tag{7.1}$$

where P_s and P_q denote the probabilities that the variable x_t assumes a value inside the s^{th} and q^{th} bin, and $P_{s,q}$ is the joint probability that x_t is in bin s and $x_{t+\tau}$ in bin q. Then τ is chosen as the first local minima of $I(\tau)$. Subsequent to this, the optimal embedding dimension m is calculated using the false nearest neighbor algorithm [193]. This algorithm assumes that the phase space of a deterministic system folds and unfolds smoothly with no sudden irregularities appearing in its structure [186]. Given these values of τ and m for a time series, we form the matrix with the i^{th} row representing i^{th} phase space vector $p(i)$. This set of m-dimensional points $p(i)$ is the reconstructed phase space.

$$X = \begin{pmatrix} x_1 & x_{\tau+1} & \dots & x_{(m-1)\tau+1} \\ x_2 & x_{\tau+2} & \dots & x_{(m-1)\tau+2} \\ \vdots & \vdots & \ddots & \end{pmatrix} \tag{7.2}$$

This embedding into m-dimensional phase space recreates the dynamics of the system. This is then represented using the metric and dynamical *invariants* of the system – the Lyapunov exponent, correlation integrals and correlation dimension. These *invariants* of a system's attractor quantify the properties that are invariant

under smooth transformations of the phase space.

The Lyapunov exponent characterizes the level of chaos in the system. It describes the dynamics of a trajectory evolution by characterizing the average rate of convergence or divergence of two neighboring trajectories in the phase space. Positive exponents mean that the trajectories are diverging and hence is a representative of the chaotic dynamics. It quantifies the sensitive dependence on initial conditions by showing the average rate at which two close points separate with time. To calculate the Lyapunov exponent, we first estimate the maximum divergence around an arbitrarily chosen reference point $p(i)$. We then determine all the points $p(k)$ which are within ϵ distance of $p(i)$. These neighboring points are used as the starting point of nearby trajectories. Then the average distance of all these trajectories to the reference trajectory is computed as a function of relative time Δn.

$$D_i(\Delta n) = \frac{1}{r}\sum_{s=1}^{r} |x_{k+(m-1)\tau+\Delta n} - x_{i+(m-1)\tau+\Delta n}| \tag{7.3}$$

where s counts the number of neighboring points of $p(i)$. There are r such neighboring points that fulfill $||p(i) - p(k)|| < \epsilon$. This $ln(D_i(\Delta n))$ is then averaged over c such arbitrarily chosen reference points.

$$S(\Delta n) = \frac{1}{c}\sum_{i=1}^{c} ln(D_i(\Delta n)) \tag{7.4}$$

The maximal Lyapunov exponent is then calculated as the slope of this graph $S(\Delta n)$ versus (Δn).

Correlation integrals and correlation dimension, on the other hand, give an estimate of the system complexity. Correlation integral $C(\epsilon)$ is a metric invariant

which characterizes the density of points in the phase space. This is done by calculating the fraction of pairs of points in the reconstructed phase space that are in the ϵ neighborhood of each other.

$$C(\epsilon) = \frac{2}{N(N-1)} \sum_{s=1}^{N} \sum_{t=s+1}^{N} H(\epsilon - ||p(t) - p(s)||) \tag{7.5}$$

where H is the Heaviside function. Correlation dimension is then calculated as the slope of the regression of $log\ C(\epsilon)$ versus $log\ \epsilon$. As we show in the experiments, these invariants serve as strong discriminative features. It is important to note that these descriptors of the dynamics have been extracted directly from the given time series without making specific assumptions about the system.

7.5.1.1 Implementation Details

In actual experiments, we first extract the 960-dimensional Gist descriptor per video-frame. Each of the dimensions in the sequence thus obtained is considered to be an individual time-series. For each of these time-series, the chaotic invariants are computed. These chaotic invariants include Lyapunov exponent, correlation dimension and correlation integral $C(\epsilon)$ for 8 values of ϵ. These forms a 10 dimensional vector for each time series and hence provides a 9600 dimensional vectorial representation of each video.

7.6 Experiments

We now demonstrate through experiments how capturing scene motion can be effectively used to improve dynamic scene recognition. We also show how videos

Figure 7.2: Dynamic Scene Dataset consisting of 13 classes with 10 videos per class. Top 2 rows show an example from 12 out of the 13 classes in the order Avalanche, Iceberg Collapse, Landslide, Volcano eruption, Chaotic traffic, Smooth traffic, Forest fire, Waterfall, Boiling water, Fountain, Waves and Whirlpool. The bottom row shows frames from 6 videos of the 13^{th} class Tornado. Notice the large intra-class variation in the dataset. Large variations in the background, illumination, scale and view can be easily seen. Similar variations are present in each of the classes.

can be labeled with motion-attributes derived from the scene-dynamics. First, we briefly describe the dataset that is used.

7.6.1 Dataset

Due to the lack of a suitable dataset, we compiled a dataset of 13 classes with 10 videos per class. Each of these videos has been downloaded from video hosting websites like 'Youtube'. As there was no control over the video capturing process, the dataset has large variations in terms of illumination, rate, view and scale. Also, there is a variation in resolution and camera dynamics. These variations

along with the underlying physics of the process have ensured that the intra-class variation is very high. Various classes in this dataset are: Avalanche, Boiling Water, Chaotic Traffic, Forest Fire, Fountain, Iceberg Collapse, Landslide, Smooth Traffic, Tornado, Volcanic Eruption, Waterfall, Waves and Whirlpool. Figure 7.2 shows an example frame from one randomly chosen video of each class. The last row in this figure shows 6 examples from the 13^{th} class 'Tornado'. Notice the large intra-class variation in the classes. This dataset can be downloaded from the project page http://www.umiacs.umd.edu/users/nshroff/DynamicScene.html

Before we proceed with the experiments, we note that pure static appearance based classification on randomly chosen frames from these videos will cause confusion between classes such as {'chaotic traffic' and 'smooth traffic'}, {'avalanche' and 'iceberg'},{'waterfall' and 'fountain'},{'landslide' and 'volcanic eruption'},{'whirlpool' and 'waves'}. Further, using only the dynamics of the scene will cause a confusion between classes like {'avalanche','landslide','volcanic eruptions'},{'tornado' and 'whirlpool'},{'boiling water' and 'forest fire'}.

7.6.2 Recognizing Dynamic Scenes in the Wild

In this section, we present experiments for dynamic scene recognition performed on the dataset described in section 7.6.1. We compare the performance of the chaos framework with well-known methods – LDS model and a bag-of-words (BoW) model. We restrict ourselves to these methods, as they are also 'global' in nature and do not require segmentation of the scene into objects. We first compare

the performance using a NN classifier. This serves as a transparent method for comparing various algorithms without any impact of hidden tunable parameters such as in the case of Support Vector Machines (SVMs). Next, we show the improvement obtained by using SVMs. In the following, we describe the implementation details of the other methods against which we compare.

Linear Dynamic Systems: [185] is a parametric model for spatio-temporal data and can be represented by:

$$x(t+1) = Ax(t) + w(t) \;\; w(t) \sim N(0, R) \tag{7.6}$$

$$z(t) = Cx(t) + v(t) \;\; v(t) \sim N(0, Q) \tag{7.7}$$

where $x(t)$ is the hidden state vector, $z(t)$ is the observation vector, $w(t)$ and $v(t)$ are noise components and modeled as normal with 0 mean and covariance R and Q respectively. Here, A is state-transition matrix and C is the observation matrix. Let

$$[z(1), z(2), ...z(\tau)] = U\Sigma V^T$$

be the singular value decomposition of the data matrix for τ observations. Then the model parameters are calculated [185] as $\hat{C} = U$ and

$$\hat{A} = \Sigma V^T D_1 V (V^T D_2 V)^{-1} \Sigma^{-1}$$

where

$$D_1 = [0\ 0;\ I_{\tau-1}\ 0]$$

and

$$D_2 = [I_{\tau-1}\ 0;\ 0\ 0]$$

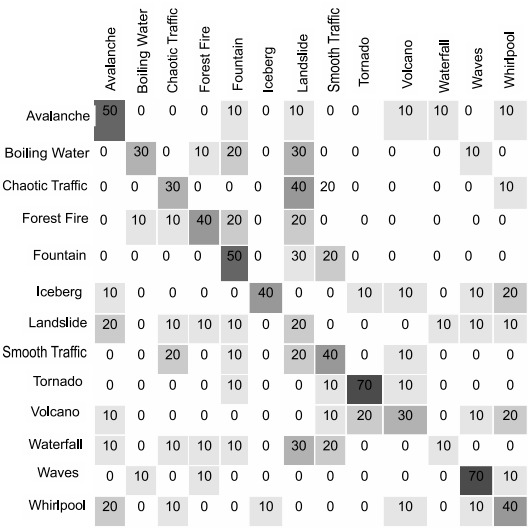
The distance metric used was based on subspace angles [194]. As a baseline, we first use the per frame intensity image as the feature vector. This gives a low performance – 13% – due to the large unconstrained photometric variations. Therefore, to better understand the performance of this model, we also use global scene features that are relatively invariant to photometric changes. In this case, we used the GIST [176] [1] descriptor extracted from each frame of the video as the observation vector and use it to learn the LDS model.

Bag-of-words: In this formalism, a video is represented as a vector in an N-dimensional Euclidean space by using a N-bin histogram of visual words. These words are learnt by clustering descriptors extracted around interest points [3] [2] from the video. These local patches are then used to build a global representation of the video. These patches were extracted for each video with the detection parameters set to $\sigma = 2$ and $\tau = 2.5$. We then build our codebook by clustering these flattened cuboids of intensity values into $N = 1000$ words. Each video was then represented using the histogram of these words normalized by the total number of words in the video. Variations in representing the cuboids using the covariance features [100] and gradients were also tried. Best results were obtained with intensity-based descriptors as shown in Table 7.1.

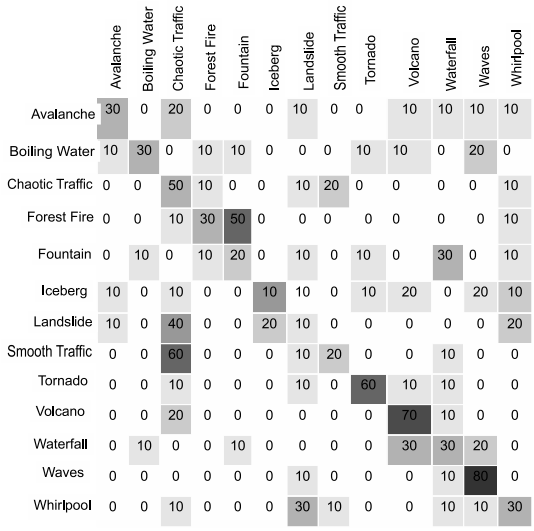
Mean Gist: We also use the mean GIST computed from the sequence of images as a global spatial descriptor of the video. This provides a 960 dimensional representation per video. The mean GIST can be seen as a simple fusion of frame-

[1] Code available at http://people.csail.mit.edu/torralba/code/spatialenvelope/

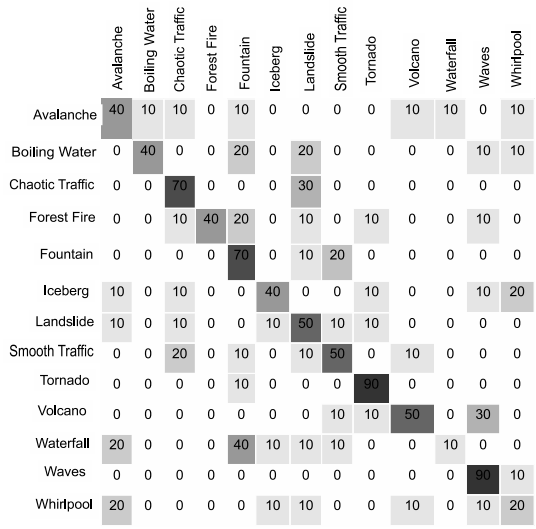
[2] Code available at http://vision.ucsd.edu/∼pdollar



(a) Static



(b) Dynamics



(c) Static + Dynamics

Figure 7.3: Confusion Tables for the three top performing algorithms. Rows represent the ground truth label and columns represent the predicted label for the given video.

wise static representations of the video.

Chaotic Invariants: As mentioned in section 7.5.1, each video was represented using a 9600 dimensional vector. This was obtained [3] by concatenating Lyapunov exponent, correlation dimension and correlation integral for 8 values of ϵ for each dimension of the GIST vector. In our implementation, the values of epsilon used were: $\{0.085, 0.05, 0.03, 0.025, 0.01, 0.008, 0.007, 0.005, 0.003,$
$0.002, 0.001\}$.

Spatio-Temporal Fusion: Chaotic invariants by themselves provide only the dynamical invariants, but do not encode the spatial appearance. Hence, we fuse the global spatial information provided by mean GIST and the dynamic information provided by the chaotic invariants into a single feature vector. This provides a simple way for fusing the spatial and temporal cues. The distance metric then uses a weighted combination of the static and global distances.

We perform a leave-one-video-out recognition experiment to compare the performance of various algorithms discussed above and the results are shown in Table 7.1. We can see that modeling dynamics using just the chaotic invariants leads to a reasonable performance. We obtain comparable performance using the mean GIST. However, the best performance is obtained by fusing the chaotic invariants and mean Gist. We see that LDS shows poor performance compared to chaotic invariants. This is because the latter makes no assumption about the model and just uses the observed data to obtain invariants. Further, since scenes have valuable

[3]Code available at http://www.physik3.gwdg.de/tstool/HTML/index.html, http://www.mpipks-dresden.mpg.de/~tisean/TISEAN_2.1/docs/indexf.html

spatial information, using just the dynamic cues would not suffice. Therefore, augmenting the static features with the dynamic features leads to significantly improved classification rates. Confusion tables for the three best performing algorithms are presented in Figure 7.3.

Improved Classification In this experiment, we study the effect of better classifiers on the recognition performance of dynamic scenes. For the best performing algorithm (static + dynamics), we perform the same experiment with a linear SVM. Figure 7.4 shows the comparative performance of this classifier with the nearest neighbor (NN) classifier. As seen, the performance improves from overall recognition of 52% (NN) to 58% (linear SVM with $C = 1000$). This suggests that better classifiers can be designed to improve the performance further, but this is not the primary goal of this chapter.

7.6.3 Attribute-based semantic organization

Attribute-based descriptions have been shown to possess useful properties such as, a) describing unknown classes, b) learning models of object categories from textual descriptions, and c) detecting unusual attributes of known classes. This has been demonstrated for face recognition [195], object recognition [196, 197] and static-scene recognition [176]. Here, we show how the motion-attributes discussed in section 7.4 can be used to provide semantically meaningful organization of videos.

Each of these dynamic attributes – Busyness, Granularity, and Regularity – can be estimated from video features using regression techniques. Although any of

Class	LDS (GIST)	Bag of Words	Mean (GIST)	Dynamics (Chaos)	Static + Dynamics
Tornado	70	10	70	60	90
Waves	40	50	70	80	90
Chaotic Traffic	10	20	30	50	70
Fountain	0	10	50	20	70
Iceberg Collapse	20	30	40	10	50
Landslide	20	40	20	10	50
Smooth Traffic	10	0	40	20	50
Volcanic Eruption	0	30	30	70	50
Avalanche	70	30	50	30	40
Boiling Water	70	0	30	30	40
Forest Fire	0	30	40	30	40
Whirlpool	20	30	40	30	40
Waterfall	0	30	10	30	10
Overall	25%	24%	40%	36%	**52%**

Table 7.1: Comparison of classification rates of different methods using the dynamic scene dataset. The recognition accuracy is improved significantly by fusing global static features (Mean GIST) and dynamic features. It is interesting to note when modeling dynamics with simplified models (LDS, Bag-of-words) the performance is worse than the static features.

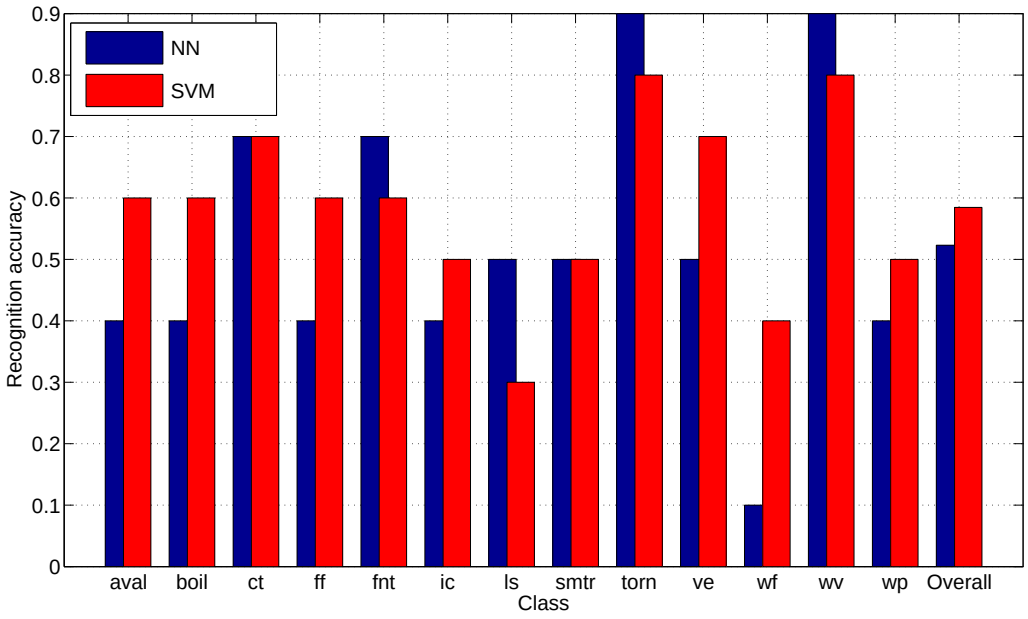


Figure 7.4: Performance comparison with two classifiers Nearest Neighbor(NN) and SVM. Shown here is the classification rates for each of the individual classes. Red bars (SVM) show an improved classification in most of the classes. The last column shows the overall performance which is improved from 52% (NN) to 58% (SVM). Figure best viewed in color.

the standard regression techniques can be used, in our experiments we use a linear regression model. Given a vector representation x of a video, the attribute value v is given by

$$v = x^T w + n$$

where w is the parameter vector. This parameter vector is then learnt by minimizing the mean squared error.

$$\hat{w} = X^+ v$$

where v is the vector of attribute values for the training videos. X is the matrix with its rows as the feature vectors of the training videos and X^+ is its pseudo-inverse. Training videos are given attribute labels from -1 (lowest) to 1 (highest). The vector v thus obtained is then used to learn the parameter vector $\hat{w}$. This is then used for predicting the attribute value $\hat{v}$ of a new video. The predicted values were then divided into 5 equal bins. In Figure 7.5(a), randomly chosen videos from each of the bins are shown. The attribute value v increases as we go down the axis. First row is a video of 'waterfall' which has only a small portion of the image in motion while the bottom row ('waves') which has large motion almost all over the image is the most busy.

A similar procedure is used for granularity and regularity attribute. Figure 7.5(b) shows similar organization over the granularity axis. The top row 'chaotic traffic' has moving structural elements that are very coarse (vehicles) whereas the bottom row 'forest fire' has very fine moving structural elements (fire). Similarly, Figure 7.6(a) shows the organization on the regularity axis. The top row shows

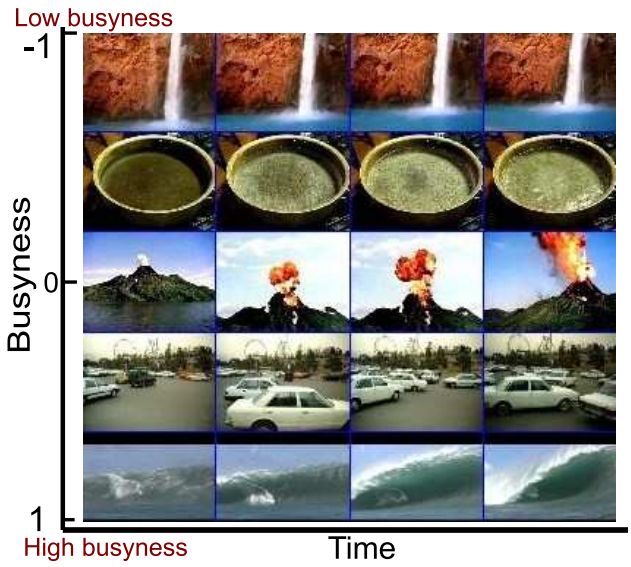
an ‘iceberg collapse’ which has a very regular gravity-driven motion in the vertical direction while the video in the bottom row ‘tornado’ exhibits motion that appears highly random, thus scores low on the ‘regularity’ axis.

We now present experiments that illustrate how scenes that share similar spatial attributes can be distinguished based on their dynamic attributes. We show that spatial attributes augmented with dynamic attributes lead to a much better separation of scenes. Consider the 2 classes ‘waves’ and ‘whirlpool’, which share very similar spatial attributes like ‘naturalness’, ‘roughness’ [176] etc. But when the dynamic attributes ‘regularity’ and ‘busyness’ are considered, a clear separation can be seen in Figure 7.7(a). Simple linear and quadratic classifiers easily separates the 2 classes. Similar separation can be seen for the case of ‘chaotic traffic’ and ‘smooth traffic’ in Figure 7.7(b) and for ‘avalanche’ and ‘iceberg’ in Figure 7.6(b).

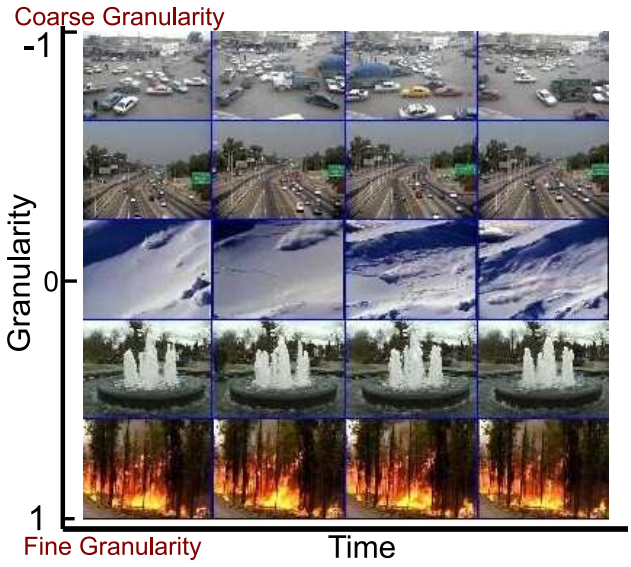
7.7 Conclusion and Future Work

In this chapter, we studied the problem of categorizing ‘in-the-wild’ dynamic scenes. We showed that capturing dynamics within scenes leads to better representation of scenes. Further, we showed that using motion attributes leads to a meaningful organization of the videos. State-of-the-art algorithms to capture dynamics seem to perform poorly.

We proposed a method based on chaotic systems that outperforms other standard techniques, but overall performance is reminiscent of the early attempts at unconstrained object recognition [198]. Results show significant improvement by

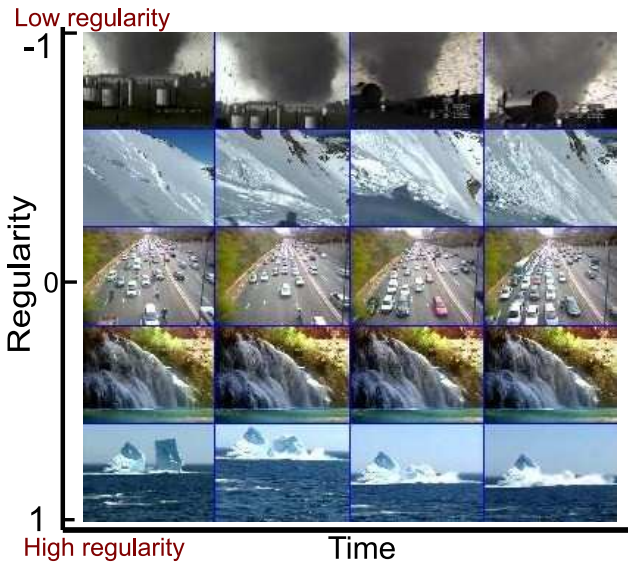


(a) Degree of Busyness

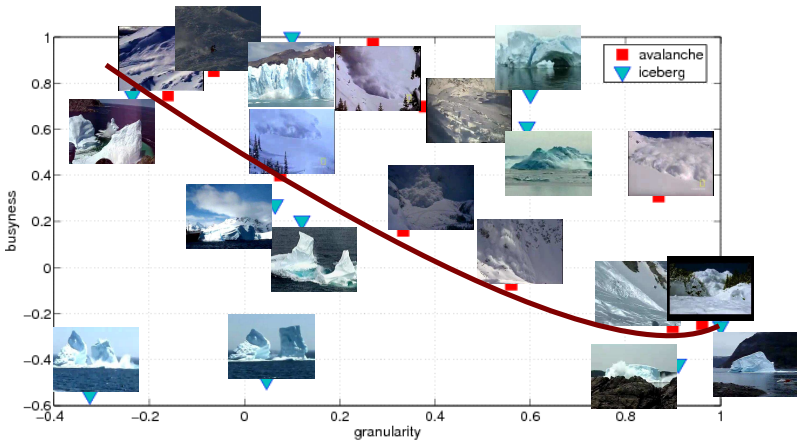


(b) Degree of Granularity

Figure 7.5: Organization of dynamic scenes by the degree of (a) busyness and (b) granularity. Videos were divided into 5 equal bins and a randomly chosen video from each bin is shown. x-axis shows the evolution over time. (a) Top row is a 'waterfall' video where the motion is not-so-busy and is confined spatially to a small part of the video. Highly busy bottom row is a 'waves' video where the motion is present over almost all of the frame. (b) Top row with coarse granularity video corresponds to the 'chaotic traffic' class whose moving elements are coarse (vehicles). While the bottom row is a video from 'forest fire' which has a very fine granularity (fire).

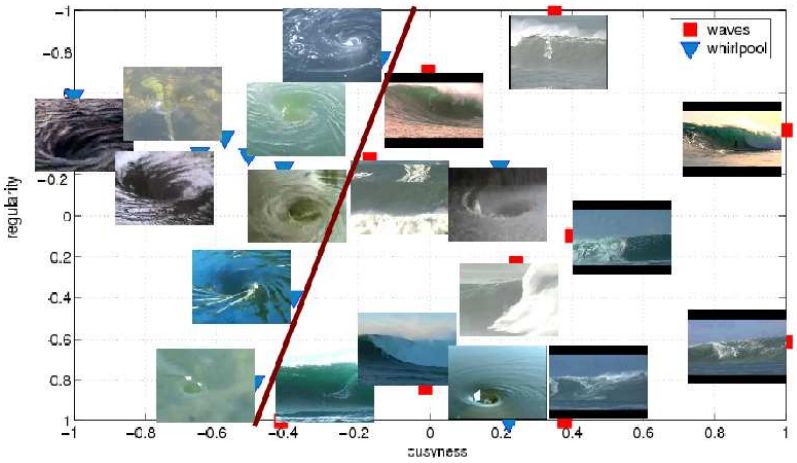


(a) Degree of regularity

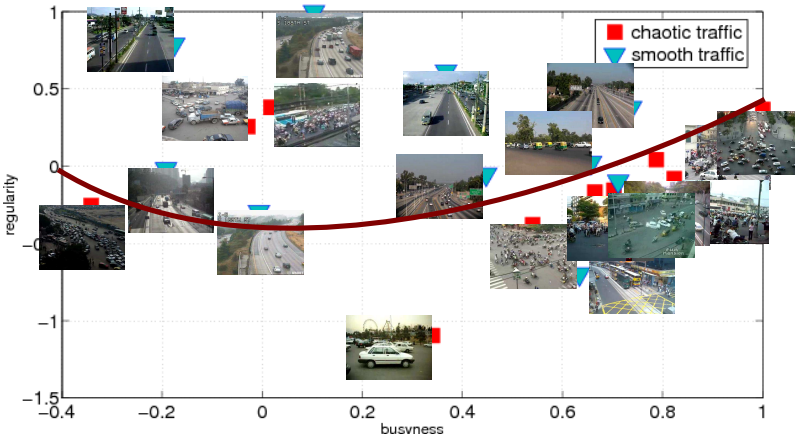


(b) 'Avalanche' vs. 'Iceberg'

Figure 7.6: (a) Organization of dynamic scenes according to the degree of regularity. Top row with very irregular and random motion is a video from the 'tornado' class. The bottom row corresponds to a video of 'iceberg' class with regular and constrained motion of an iceberg collapsing in the vertical direction under the effect of gravity. (b) Separation of the classes 'avalanche' and 'iceberg'. These 2 categories have very similar spatial attributes and are difficult to separate using them. But a clear separation (red line) can be seen between the 2 classes on the 2 temporal attributes 'granularity' and 'busyness'. It should be noted that the red line has been drawn for the sake of visual illustration. A simple quadratic classifier easily separates the 2 classes. Note that 3 videos from 'iceberg' class, although with low granularity, end up falling in the wrong side because of being highly busy.



(a) 'Waves' vs. 'Whirlpool'



(b) 'Chaotic' vs. 'Smooth' Traffic

Figure 7.7: Separation of the spatially similar classes using motion attributes. Both pairs of classes have very similar spatial attributes and thus are difficult to separate using them. A clear separation (red line) can be seen between the 2 classes on the 2 temporal attributes 'regularity' and 'busyness'. It should be noted that the red line has been drawn for the sake of visual illustration. Simple classifiers (linear and quadratic respectively) easily separate the 2 classes. Note that in (a) 2 examples of the 'whirlpool' class and in (b) 2 examples from each class fall on the wrong side.

fusing static and dynamic features. Experimental results are promising. But, the categorization of dynamic scenes in the unconstrained setting poses a very challenging task due to its large intra-class variations and hence requires further work in this direction.

Chapter 8

Directions for Future Work

There are multiple avenues for extending the findings of this dissertation. We discuss a few below.

8.1 Optical Flow in the Presence of Varying Blur

As discussed in [199] (chapter 2 of this dissertation), two consecutive frames with different focal settings cause the brightness constancy assumption to fail. This causes spurious optical flow while estimating motion. To overcome this, we want to explore ways to robustly estimate optical flow in the presence of varying blur. Towards this direction, Myles and Lobo [200] proposed an iterative approach to simultaneously estimate motion and defocus blur. They made a restrictive assumption on motion to be affine. Seitz and Baker [201] formulated this as a Markov Random Field inference problem but again restricted to the global affine motion. We want to explore this open problem by incorporating blur into the optical flow cost. Towards this goal, we also want to explore the performance and advantage of computing multi-frame optical flow.

8.2 Bin Picking

Machine vision algorithms have great applications in industrial automation and assembly. Most of the industrial objects in assembly lines are metallic and hence have specular surfaces. In order to reliably estimate the pose of such objects, in [202] (chapter 3 of this dissertation), we proposed a novel specular feature that can be extracted very reliably on high curvature metallic objects. Developing pose estimation algorithms using thees specular features for a wider class of objects would be an interesting future direction. Another interesting direction would be to develop features that can be reliably detected on low curvature metallic objects.

8.3 Video Précis

Video summarization mainly requires one to identify (a) important objects and/or events, and (b) any 'unique' object/event present in the video. An interesting future direction would be to learn models to identify both important and anomalous objects/events automatically over time for cameras mounted at a specific location.

Recently, there has been interest in recognizing actions from videos that have been captured by wearable cameras mounted on human heads. This has been termed as the egocentric vision. So, in near future, one can imagine huge volumes of egocentric vision data being captured. Identifying important objects, persons and using it to summarize the day-long video is an interesting open problem.

8.4 Dynamic Scenes

Dynamic scenes typically have a characteristic motion of its structural elements [203]. For instance, waterfalls have vertically down motion while forest fire have typical vertically upward motion. Similarly, avalanches and landslides have sliding motion in downward direction. Understanding and characterizing the physics of this motion of scene's structural elements would lead to a better representation of the scenes. Furthermore, in unconstrained videos, camera motion is present in large number of videos. Improved modeling of the dynamics of the scene would require separating this camera motion from the motion of scene elements.

8.5 Action Detection

Spatio-temporal localization of human actions in unconstrained videos is a very challenging task. A substantial reason behind this is the lack of annotated data. Most of the existing annotations are associated with the videos as a whole rather than the spatio-temporal location of the action of interest. The problem is further aggravated by intra-class variations caused due to view, execution rate, illumination variation, occlusions, presence of multiple actions and sometimes multiple instance of the same action. Recently, there has been some interest in this direction [204, 205], but a lot more needs to be done to reliably localize actions in videos. We want to explore this challenging problem and extend it to real-time systems which implies developing an online extension of action detection i.e., detecting and labeling the action while it is being executed.

Appendix A

Summarizing Streaming Data

A.1 Introduction

In recent years, video streams have become ubiquitous because of the large number of mobile cameras or pre-installed cameras. These cameras generate huge volumes of video data and stream it to a distant user or server. Even larger amount of streaming video data is generated because of the ubiquitous nature of mobile devices accessing videos from datasets like YouTube. Depending upon the specific scenario, streaming the whole video could be expensive in terms of bandwidth, storage, time, etc. Instead, the user-experience could be improved greatly if one can obtain concise representation of the video while it is being streamed. Such a technique will also be useful for sports DVRs recording only the highlights of the live sports match. These and several other applications necessitates the requirement to develop fast, online and one-pass algorithm to summarize streaming data. Moreover, such video abstraction techniques would make summarization scalable to large video databases. But, an online summarization algorithm faces several additional challenges, in particular:

- Information about the content of the incoming video is unavailable.
- In some cases, even the length of the incoming video might be unavailable.

- Access to only the current (or last few) video segments available along with the current summary.

- Small delay restricts the amount of computations that is allowed.

For these reasons, many existing subset selection/summarization methods cannot be directly applied on video streams. Recently, this problem of generating summaries online has attracted some attention [206, 207]. As discussed in chapter 4, one line of thought for video abstraction has been first clustering data and then picking cluster centers as exemplars. In this approach, an online video summarization corresponds to the clustering of data while it is being streamed. Over the past decade, there has been few work which clusters streaming data [208, 209]. But, as discussed earlier, these clustering techniques, get skewed by uneven distribution of clusters/events. To remedy this issue, we discuss next the online précis algorithm.

A.2 Online Précis

In this section, we discuss the online extension of the algorithm 5.3. This algorithm has been summarized in algorithm A.1.

Let $X = \{x_1, x_2, \ldots, x_t, \ldots\} \in \mathcal{M}$ be the streaming data, where, x_t be the data-point received at time t. Let ν be the maximum allowed memory i.e., the maximum number of past data-points that can be stored and accessed at any given time instant. The goal here is to extract a subset E of k exemplars while the data is continuously being streamed and while satisfying the memory constraint.

We start with streaming first ν data points $V = \{x_1, x_2, \ldots, x_\nu\}$ and give it

as input to the algorithm 5.3. Here, along with the set of exemplars E, we also maintain the set of cluster centers Φ, cluster support vector $\beta \in \mathbb{N}^{1\times k}$. Similar procedure is also adopted in the online k-means algorithm.

Subsequently, another $\nu - k$ data points are streamed to form

$$V = \{E,\ \ x_{\nu+1},\ \ldots,\ x_{2\nu-k}\}$$

Now, V is first optimized for diversity. In order to optimize for coverage, we first assign the cluster centers Φ and current data points to the nearest exemplar. Now, a weighted mean is obtained for each cluster where the cluster center from the previous iteration is weighted by the cluster support β. Then, the closest exemplar to the obtained mean is used as exemplar.

Algorithm A.1: Online Precis Algorithm for Streaming Data

Input: Streaming data points $X = \{x_1, x_2, \ldots, x_t, \ldots\} \in \mathcal{M}$, Number of exemplars k, Memory Restrictions ν

Output: $E = \{e_1, \ldots e_k\} \subseteq X$

Initial setup:

Stream first ν data points $V \leftarrow \{x_1, x_2, \ldots, x_\nu\}$

Initialize Φ with k zero valued data points.

$[E \;\; \Phi \;\; \beta] \leftarrow OnlineSubsetSelection(V, k, \Phi, \beta)$ as in algorithm A.2

$V \leftarrow E$

while $t \leq n$ **do**

  for $\gamma \leftarrow 1$ **to** $\nu - k$ **do**
    $V \leftarrow \{V, \;\; x_t\}$

  end

  $[E \;\; \Phi \;\; \beta] \leftarrow OnlineSubsetSelection(V, k, \Phi, \beta)$ as in algorithm A.2

  $V \leftarrow E$

end

Algorithm A.2: Online Subset selection

Input: Data points $V = \{x_1, x_2, \ldots, x_n\} \in \mathcal{M}$, Number of exemplars k,

Cluster Means Φ, Cluster Support β

Output: $E = \{e_1, \ldots, e_k\}$, Cluster Means ζ, Cluster Support β

$tol \leftarrow 1$

Diversity: $E \leftarrow \text{Div}(V, k, tol)$ as in algorithm 5.2.

`// Assign mean of every cluster (`Φ_i`) to the closest exemplar`

for $i \leftarrow 1$ **to** k **do**

$\omega_i \leftarrow arg\ min_j\ d(\Phi_i, e_j)$

end

`// Assign every data point to the closest exemplar`

for $i \leftarrow 1$ **to** n **do**

$\alpha_i \leftarrow arg\ min_j\ d(v_i, e_j)$

end

for $i \leftarrow 1$ **to** k **do**

`// Compute weighted mean of individual clusters`

$\zeta_i \leftarrow WeightedMean\big(\Phi(\omega == i), \beta(\omega == i), V(\alpha == i), e_i\big)$

`// Choose data-point closest to the mean as exemplars`

$idx \leftarrow\ arg\ min_j\ d(v_j, \zeta_i)$

$e_i \leftarrow v_{idx}$

end

Update cluster support β

Appendix B

Geodesic Computation on Grassmann manifold

The Grassmann manifold can be interpreted as a quotient of the special orthogonal group $SO(m)$. For a point $\zeta \in SO(m)$, the geodesic flow in a tangent direction is given by $\psi_\zeta(A,t) = \zeta^T \exp(tA)$, where exp is the matrix exponential. This can be used to deduce the expression for the geodesic flow on $\mathcal{G}_{m,p}$. Geodesic flow starting from a point $\Delta = \zeta^T J$ is of the type: $t \rightarrow \exp(tA)J$, where A takes the form $\begin{bmatrix} 0 & -B \\ B^T & 0 \end{bmatrix}$ and $J = \begin{bmatrix} I_{p\times p} \\ 0 \end{bmatrix}$. The analytical expression for the inverse exponential map on the Grassmann manifold is unavailable and is computed numerically. Efficient algorithm to compute the inverse exponential map for the Grassmann manifold $\mathcal{G}_{m,p}$ was proposed by [143]. This algorithm has been described in Algorithm B.1

Algorithm B.1: Algorithm for computation of Inverse Exponential Map on Grassmann Manifold $\mathcal{G}_{m,p}$

Input: X_0, X_1 on the manifold $\mathcal{G}_{m,p}$

Output: A

Compute the $m \times m$ orthogonal completion O of X_0

Compute Thin CS decomposition [210] of $O^T X_1$ given by

$$O^T X_1 = \begin{pmatrix} Y \\ Z \end{pmatrix} = \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \\ 0 \end{pmatrix} S_1^T = \begin{pmatrix} V_1 & 0 \\ 0 & \tilde{V}_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \end{pmatrix} S_1^T \tag{B.1}$$

$\gamma_i \leftarrow i^{th}$ diagonal element of Γ

$\sigma_i \leftarrow i^{th}$ diagonal element of Σ

Compute θ_i such that $\gamma_i = \cos(\theta_i)$ and $\sigma_i = \sin(\theta_i)$

$$\Theta \leftarrow \begin{pmatrix} \theta_1 & 0 & 0 & \dots \\ 0 & \theta_2 & 0 & \dots \\ \vdots & \vdots & \ddots & \end{pmatrix}$$

$A \leftarrow \tilde{V}_2 \Theta V_1$

Bibliography

[1] P. Turaga and R. Chellappa, "Nearest-neighbor search algorithms on non-euclidean manifolds for computer vision applications," in *ICVGIP*, 2010.

[2] Y. Wang, H. Jiang, M. Drew, Z. Li, and G. Mori, "Unsupervised discovery of action classes," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, NY, USA, June 2006, pp. 1654–1661.

[3] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, China, Oct. 2005, pp. 65–72.

[4] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *Proceedings of International Conference on Pattern Recognition*, Cambridge, UK, Aug. 2004, pp. 32–36.

[5] S. Ullman, *The interpretation of visual motion*. MIT Press, 1979.

[6] B. Rogers and M. Graham, "Motion parallax as an independent cue for depth perception." *Perception*, vol. 8, no. 2, pp. 125–134, 1979.

[7] J. Gibson, P. Olum, and F. Rosenblatt, "Parallax and perspective during aircraft landings," *The American Journal of Psychology*, vol. 68, no. 3, pp. 372–385, 1955.

[8] K. Nakayama and J. M. Loomis, "Optical velocity patterns, velocity-sensitive neurons, and space perception: a hypothesis," *Perception*, vol. 3, no. 1, pp. 63–80, 1974.

[9] J. Koenderink and A. J. Van Doorn, "How an ambulant observer can construct a model of the environment from the geometrical structure of the visual inflow," in *Kybernetik*, G. Hauske and E. Butenandt, Eds. Oldenburg, 1977, pp. 224–247.

[10] J. Todd, "The perception of three-dimensional structure from rigid and non-rigid motion," *Perception and Psychophysics*, vol. 36, no. 2, pp. 97–103, 1984.

[11] S. Ullman, "The interpretation of structure from motion," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 203, no. 1153, pp. 405 – 426, 1979.

[12] T. Broida and R. Chellappa, "Estimating the kinematics and structure of a rigid object from a sequence of monocular images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 497–513, 1991.

[13] R. Hartley and A. Zisserman, *Multiple view geometry*. Cambridge university press Cambridge, UK, 2000.

[14] M. Braunstein and G. Anderson, "Shape and depth perception from parallel projections of three-dimensional motion," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 10, no. 6, pp. 749–760, 1984.

[15] N. Rubin and N. Albert, "Real-world scenes can be easily recognized from their edge-detected renditions: Just add motion!" *Journal of Vision*, vol. 1, no. 3, p. 37, 2001.

[16] G. Pike, R. Kemp, N. Towell, and K. Phillips, "Recognizing moving faces: The relative contribution of motion and perspective view information," *Visual Cognition*, vol. 4, no. 4, pp. 409–438, 1997.

[17] J. D. Courtney, "Automatic video indexing via object motion analysis," *Pattern Recognition*, vol. 30, no. 4, pp. 607 – 625, 1997.

[18] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 195–202, June 2003.

[19] M. Subbarao and G. Surya, "Depth from defocus: a spatial domain approach," *International Journal of Computer Vision*, vol. 13, no. 3, pp. 271–294, 1994.

[20] A. Pentland, S. Scherock, T. Darrell, and B. Girod, "Simple range cameras based on focal error," *Journal of the Optical Society of America A*, vol. 11, no. 11, pp. 2925–2934, 1994.

[21] S. Chaudhuri and A. Rajagopalan, *Depth from defocus: a real aperture imaging approach*. Springer Verlag, 1999.

[22] A. N. Rajagopalan and S. Chaudhuri, "Optimal recovery of depth from defocused images using an MRF model," in *ICCV*, 1998.

[23] P. Favaro and S. Soatto, "A geometric approach to shape from defocus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 406–417, 2005.

[24] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *CVPR*, vol. 1, June 2003, pp. 195–202.

[25] E. Dowski and W. Cathey, "Extended depth of field through wave-front coding," *Appl. Opt.*, vol. 34, pp. 1859–1866, 1995.

[26] A. Veeraraghavan, R. Raskar, A. Agrawal, A. Mohan, and J. Tumblin, "Dappled photography: Mask enhanced cameras for heterodyned light fields and coded aperture refocusing," *ACM Transactions on Graphics*, vol. 26, no. 3, 2007.

[27] A. Levin, R. Fergus, F. Durand, and B. Freeman, "Image and depth from a conventional camera with a coded aperture," *ACM Transactions on Graphics*, 2007.

[28] H. Nagahara, S. Kuthirummal, C. Zhou, and S. K. Nayar, "Flexible Depth of Field Photography," in *ECCV*, 2008, pp. 60–73.

[29] S. W. Hasinoff, K. N. Kutulakos, F. Durand, and W. T. Freeman, "Time-constrained photography," in *ICCV*, 2009, pp. 333–340.

[30] B. Horn and B. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, pp. 185–203, 1981.

[31] M. Subbarao and G. Surya, "Application of spatial-domain convolution/deconvolution transform for determining distance from image defocus," in *SPIE*, 1992.

[32] P. Favaro, "Recovering thin structures via nonlocal-means regularization with application to depth from defocus," in *CVPR*, 2010.

[33] X. Ren and J. Malik, "Learning a classification model for segmentation," in *ICCV*, 2003.

[34] C. Zitnick, N. Jojic, and S. Kang, "Consistent segmentation for optical flow estimation," in *ICCV*, 2005, pp. 1308–1315.

[35] Y. Taguchi, B. Wilburn, and L. Zitnick, "Stereo reconstruction with mixed pixels using adaptive over-segmentation," in *CVPR*, 2008, p. 18.

[36] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen, "Interactive digital photomontage," in *SIGGRAPH*, 2004.

[37] Y. Boykov, O. Veksler, and R. Zabhi, "Efficient Approximate Energy Minimization via Graph Cuts," *IEEE transactions on PAMI*, vol. 20, no. 12, pp. 1222–1239, Nov. 2001.

[38] A. Adams, D. Jacobs, J. Dolson, M. Tico, K. Pulli, E. Talvala, B. Ajdin, D. Vaquero, H. Lensch, M. Horowitz *et al.*, "The Frankencamera: an experimental platform for computational photography," in *ACM SIGGRAPH 2010 papers*. ACM, 2010, pp. 1–12.

[39] A. Levin, P. Sand, T. Cho, F. Durand, and W. Freeman, "Motion-invariant photography," in *ACM SIGGRAPH 2008 papers*. ACM, 2008, pp. 1–9.

[40] S. Hasinoff and K. Kutulakos, "Confocal stereo," *International journal of computer vision*, vol. 81, no. 1, pp. 82–104, 2009.

[41] R. Raskar, K.-H. Tan, R. Feris, J. Yu, and M. Turk, "Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 679–688, 2004.

[42] E. P. Degarmo, J. T. Black, and R. A. Kohser, *Materials and Processes in Manufacturing (9th ed.)*. Wiley, 2003.

[43] B. Horn and K. Ikeuchi, "Picking Parts Out of a Bin," MIT Artificial Intelligence Laboratory, Tech. Rep. AIM-746, 1983.

[44] K. Rahardja and A. Kosaka, "Vision-based bin-picking: recognition and localization of multiple complex objects using simple visual cues," in *IROS*, 1996.

[45] A. Agrawal, Y. Sun, J. Barnwell, and R. Raskar, "Vision-guided Robot System for Picking Objects by Casting Shadows," *The International Journal of Robotics Research*, vol. 29, no. 2–3, pp. 155–173, 2010.

[46] D. G. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artificial Intelligence*, vol. 31, no. 3, pp. 355–395, 1987.

[47] ——, "Fitting parameterized three-dimensional models to images," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 441–450, 1991.

[48] D. DeMenthon and L. S. Davis, "Model-based object pose in 25 lines of code," in *ECCV*, 1992, pp. 335–343.

[49] P. F. Felzenszwalb and J. D. Schwartz, "Hierarchical matching of deformable shapes," in *CVPR*, 2007, pp. 1–8.

[50] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[51] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, R. Chellappa, A. Agrawal, and H. Okuda, "Pose estimation in heavy clutter using a multi-flash camera," in *ICRA*, 2010.

[52] R. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 3–26, 1986.

[53] S. Wang, R. Cromwell, A. Kak, I. Kimura, and M. Osada, "Model-based vision for robotic manipulation of twisted tubular parts: using affine transforms and heuristic search," in *ICRA*, 1994.

[54] G. Brelstaff and A. Blake, "Detecting specular reflections using Lambertian constraints," in *ICCV*, 1988.

[55] S. Nayar, X.-S. Fang, and T. Boult, "Removal of Specularities using Color and Polarization," in *CVPR*, 1993.

[56] S. Mallick, T. Zickler, D. Kriegman, and P. Belhumeur, "Beyond Lambert: Reconstructing specular surfaces using color," in *CVPR*, 2005.

[57] J. Chang, R. Raskar, and A. Agrawal, "3D pose estimation and segmentation using specular cues," in *CVPR*, 2009.

[58] G. Healey and T. Binford, "Local shape from specularity," *Computer Vision, Graphics, and Image Processing*, vol. 42, no. 1, pp. 62–86, 1988.

[59] K. Gremban and K. Ikeuchi, "Planning multiple observations for object recognition," *International Journal of Computer Vision*, vol. 12, no. 2, pp. 137–172, 1994.

[60] A. C. Sankaranarayanan, A. Veeraraghavan, O. Tuzel, and A. Agrawal, "Image Invariants for Smooth Reflective Surfaces," in *ECCV*, 2010.

[61] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Graphics and Image Processing*, vol. 24, no. 6, pp. 381–395, 1981.

[62] R. Hartley and A. Zisserman, *Multiple view geometry*. Cambridge university press Cambridge, UK, 2000.

[63] B. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 3, no. 1, pp. 1–37, Feb. 2007.

[64] Y. Li, T. Zhang, and D. Tretter, "An overview of video abstraction techniques," HP Laboratory/HPL-2001-191, Tech. Rep., July 2001.

[65] A. Money and H. Agius, "Video summarisation: A conceptual framework and survey of the state of the art," *Journal of Visual Communication and Image Representation*, vol. 19, no. 2, pp. 121–143, Feb. 2008.

[66] X. Zhu, A. Elmagarmid, X. Xue, L. Wu, and A. Catlin, "InsightVideo: toward hierarchical video content organization for efficient browsing, summarization and retrieval," *IEEE Transactions on Multimedia*, vol. 7, no. 4, pp. 648–666, Aug. 2005.

[67] J. Oh, Q. Wen, S. Hwang, and J. Lee, "Video abstraction," in *Video data management and information retrieval*, S. Deb, Ed. Idea Group Inc. and IRM Press, 2004, pp. 321–346.

[68] C. Taskiran and E. Delp, "Video summarization," in *Digital Image Sequence Processing, Compression, and Analysis*, T. Reed, Ed. CRC Press, 2005, pp. 215–231.

[69] B. Shahraray and D. Gibbon, "Automatic generation of pictorial transcripts of video programs," in *Proceedings of Society of Photo-Optical Instrumentation Engineers*, vol. 2417, San Jose, CA, USA, Feb. 1995, pp. 512–518.

[70] S. Smoliar and H. Zhang, "Content-based video indexing and retrieval," *IEEE Multimedia Magazine*, vol. 1, no. 2, pp. 62–72, Summer 1994.

[71] H. Zhang, J. Wu, D. Zhong, and S. Smoliar, "An integrated system for content-based video retrieval and browsing," *Pattern Recognition*, vol. 30, no. 4, pp. 643–658, April 1997.

[72] H. Zhang, C. Low, S. Smoliar, and J. Wu, "Video parsing, retrieval and browsing: an integrated and content-based solution," in *Proceedings of ACM International Conference on Multimedia*, San Francisco, CA, USA, Nov. 1995, pp. 15–24.

[73] B. Gunsel, Y. Fu, and A. Tekalp, "Hierarchical temporal video segmentation and content characterization," in *Proceedings of Society of Photo-Optical Instrumentation Engineers*, vol. 3229, Dallas, TX, USA, Nov. 1997, pp. 46–56.

[74] J. Nam and A. Tewfik, "Video abstract of video," in *IEEE Third Workshop on Multimedia Signal Processing*, Copenhagen, Denmark, Sep. 1999, pp. 117–122.

[75] A. Divakaran, R. Radhakrishnan, and K. Peker, "Video summarization using descriptors of motion activity: A motion activity based approach to key-frame extraction from video shots," *Journal of Electronic Imaging*, vol. 10, no. 4, pp. 909–916, Oct. 2001.

[76] A. Divakaran, K. Peker, R. Radhakrishnan, Z. Xiong, and R. Cabasson, "Video summarization using mpeg-7 motion activity and audio descriptors," in *Video Mining*, A. Rosenfeld, D. Doermann, and D. DeMenthon, Eds. Kluwer Academic Publishers, Oct. 2003, p. 91.

[77] Z. Xiong, R. Radhakrishnan, and A. Divakaran, "Generation of sports highlights using motion activity in combination with a common audio feature extraction framework," in *Proceedings of IEEE International Conference on Image Processing*, vol. 1, Barcelona, Spain, Sept. 2003, pp. 5–8.

[78] B. Chen, J. Wang, and J. Wang, "A novel video summarization based on mining the story-structure and semantic relations among concept entities," *IEEE Transactions on Multimedia*, vol. 11, no. 2, pp. 295–312, Feb. 2009.

[79] D. DeMenthon, V. Kobla, and D. Doermann, "Video summarization by curve simplification," in *Proceedings of ACM International Conference on Multimedia*, Bristol, UK, Sep. 1998, pp. 211–218.

[80] A. Ferman and A. Tekalp, "Multiscale content extraction and representation for video indexing," in *Proceedings of Society of Photo-Optical Instrumentation Engineers*, vol. 3229, Dallas, TX, USA, Nov. 1997, pp. 23–31.

[81] Y. Zhuang, Y. Rui, T. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *Proceedings of International Conference on Image Processing*, vol. 1, Chicago, IL, USA, Oct. 1998, pp. 866–870.

[82] A. Hanjalic and H. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1280–1289, Dec. 1999.

[83] L. Zelnik-Manor and M. Irani, "Event-Based Analysis of Video," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, Kauai, HI, USA, Dec. 2001, pp. 123–130.

[84] P. Turaga, A. Veeraraghavan, and R. Chellappa, "Unsupervised view and rate invariant clustering of video sequences," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 353–371, Mar. 2009.

[85] L. Xie, P. Xu, S. Chang, A. Divakaran, and H. Sun, "Structure analysis of soccer video with domain knowledge and hidden Markov models," *Pattern Recognition Letters*, vol. 25, no. 7, pp. 767–775, May 2004.

[86] D. Zhong, R. Kumar, and S. Chang, "Real-time personalized sports video filtering and summarization," in *Proceedings of ACM International Conference on Multimedia*, vol. 9, Ottawa, Canada, Oct. 2001, pp. 623–625.

[87] R. Radhakrishnan, A. Divakaran, Z. Xiong, and I. Otsuka, "A content-adaptive analysis and representation framework for audio event discovery from "unscripted" multimedia," *EURASIP Journal on Applied Signal Processing*, vol. 2006, pp. 1–24, Jan. 2006.

[88] M. Irani, P. Anandan, and S. Hsu, "Mosaic based representations of video sequences and their applications," in *Proceedings of IEEE International Conference on Computer Vision*, Cambridge, MA, USA, June 1995, pp. 605–611.

[89] J. Wang and H. Adelson, "Representing moving images with layers," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 625–638, Sep. 1994.

[90] N. Vasconcelos and A. Lippman, "A spatiotemporal motion model for video summarization," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, USA, June 1998, pp. 361–366.

[91] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg, "Webcam synopsis: peeking around the world," in *Proceedings of IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, Oct. 2007, pp. 1–8.

[92] M. Rubinstein, A. Shamir, and S. Avidan, "Improved seam carving for video retargeting," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 27, no. 3, pp. 1–9, Aug. 2008.

[93] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, AK, USA, June 2008, pp. 1–8.

[94] Z. Li, G. Schuster, and A. Katsaggelos, "Minmax optimal video summarization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 10, pp. 1245–1256, Oct. 2005.

[95] I. Mani and M. Maybury, *Advances in Automatic Text Summarization*. MIT Press, 1999.

[96] K. Liu, E. Terzi, and T. Grandison, "ManyAspects: a system for highlighting diverse concepts in documents," *Proceedings of VLDB Endowment*, vol. 1, no. 2, pp. 1444–1447, Aug. 2008.

[97] J. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *International Journal of Computer Vision*, vol. 79, no. 3, pp. 299–318, Sep. 2008.

[98] A. Bobick and J. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, Mar. 2001.

[99] S. Soatto, G. Doretto, and Y. Wu, "Dynamic textures," in *Proceedings of IEEE International Conference on Computer Vision*, vol. 2, Vancouver, BC, Canada, July 2001, pp. 439–446.

[100] O. Tuzel, F. Porikli, and P. Meer, "Region Covariance: A fast descriptor for detection and classification," in *Proceedings of IEEE European Conference on Computer Vision*, vol. 2, Graz, Austria, May 2006, pp. 589–600.

[101] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, pp. 972–976, Feb. 2007.

[102] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[103] I. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA, Aug. 2004, pp. 551–556.

[104] N. Shroff, P. Turaga, and R. Chellappa, "Video Précis: Highlighting diverse aspects of videos," *IEEE Transactions on Multimedia*, vol. 12, no. 8, pp. 853 –868, Dec. 2010.

[105] I. Simon, N. Snavely, and S. Seitz, "Scene summarization for online image collections," in *ICCV*, 2007.

[106] W. Cochran, *Sampling techniques*. Wiley, 1977.

[107] Y. Yue and T. Joachims, "Predicting diverse subsets using structural svms," in *ICML*, 2008.

[108] J. Carbonell and J. Goldstein, "The use of mmr, diversity-based reranking for reordering documents and reproducing summaries," in *SIGIR*, 1998.

[109] M. Gu and S. Eisenstat, "Efficient algorithms for computing a strong rank-revealing QR factorization," *SIAM Journal on Scientific Computing*, vol. 17, no. 4, pp. 848–869, 1996.

[110] P. Drineas, M. Mahoney, and S. Muthukrishnan, "Relative-error CUR matrix decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, pp. 844–881, 2008.

[111] C. Boutsidis, M. Mahoney, and P. Drineas, "An improved approximation algorithm for the column subset selection problem," in *SODA*, 2009.

[112] D. Kendall, "Shape manifolds, Procrustean metrics and complex projective spaces," *Bulletin of London Mathematical society*, vol. 16, pp. 81–121, 1984.

[113] J. D. Lafferty and G. Lebanon, "Diffusion kernels on statistical manifolds," *Journal of Machine Learning Research*, vol. 6, pp. 129–163, 2005.

[114] P. T. Fletcher, C. Lu, S. M. Pizer, and S. C. Joshi, "Principal geodesic analysis for the study of nonlinear statistics of shape," *IEEE Transactions on Medical Imaging*, vol. 23, no. 8, pp. 995–1005, August 2004.

[115] A. Srivastava, S. H. Joshi, W. Mio, and X. Liu, "Statistical shape analysis: Clustering, learning, and testing," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 4, 2005.

[116] G. Golub, "Numerical methods for solving linear least squares problems," *Numerische Mathematik*, vol. 7, no. 3, pp. 206–216, 1965.

[117] T. Chan, "Rank revealing QR factorizations," *Linear Algebra and Its Applications*, vol. 88, pp. 67–82, 1987.

[118] A. Frieze, R. Kannan, and S. Vempala, "Fast Monte-Carlo algorithms for finding low-rank approximations," *Journal of the ACM (JACM)*, vol. 51, no. 6, pp. 1025–1041, 2004.

[119] A. Deshpande and L. Rademacher, "Efficient volume sampling for row/column subset selection," in *Foundations of Computer Science (FOCS)*, 2010.

[120] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*. Society for Industrial and Applied Mathematics, 2007.

[121] I. Dhillon and D. Modha, "Concept decompositions for large sparse text data using clustering," *Machine learning*, vol. 42, no. 1, pp. 143–175, 2001.

[122] R. Subbarao and P. Meer, "Nonlinear mean shift for clustering over analytic manifolds," in *CVPR*, 2006.

[123] A. Goh and R. Vidal, "Clustering and dimensionality reduction on riemannian manifolds," in *CVPR*, 2008.

[124] A. Srivastava, W. Mio, E. Klassen, and S. Joshi, "Geometric analysis of continuous, planar shapes," *Proc. 4th International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2003.

[125] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on riemannian manifolds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1713–1727, 2008.

[126] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 728–735, 2006.

[127] X. Pennec, P. Fillard, and N. Ayache, "A riemannian framework for tensor computing," *International Journal of Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.

[128] A. Veeraraghavan, A. Srivastava, A. K. Roy Chowdhury, and R. Chellappa, "Rate-invariant recognition of humans and their activities," *IEEE Trans. on Image Processing*, vol. 18, no. 6, pp. 1326–1339, June 2009.

[129] T. K. Kim, J. Kittler, and R. Cipolla, "Discriminative learning and recognition of image set classes using canonical correlations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1005–1018, June 2007.

[130] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *International Conference on Machine Learning*, June 2008, pp. 376–383.

[131] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell, "Face recognition with image sets using manifold density divergence," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2005, pp. 581–588.

[132] Y. M. Lui and J. R. Beveridge, "Grassmann registration manifolds for face recognition," in *ECCV*, 2008.

[133] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, May 2005.

[134] Y. M. Lui, J. R. Beveridge, and M. Kirby, "Canonical stiefel quotient and its application to generic face recognition in illumination spaces," in *Biometrics: Theory, Applications, and Systems*, August 2009.

[135] A. Srivastava and X. Liu, "Tools for application-driven linear dimension reduction," *Neurocomputing*, vol. 67, pp. 136–160, 2005.

[136]

[137] P. K. Turaga, A. Veeraraghavan, and R. Chellappa, "Statistical analysis on stiefel and grassmann manifolds with applications in computer vision," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[138] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Communications on Pure and Applied Mathematics*, vol. 30, no. 5, pp. 509–541, 1977.

[139] X. Pennec, "Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements," *Journal of Mathematical Imaging and Vision*, vol. 25, no. 1, pp. 127–154, 2006.

[140] T. Lin and H. Zha, "Riemannian manifold learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 796–809, 2008.

[141] A. Trouvé, "Diffeomorphisms groups and pattern matching in image analysis," *International Journal of Computer Vision*, vol. 28, pp. 213–221, July 1998.

[142] W. Mio, A. Srivastava, and S. Joshi, "On shape of plane elastic curves," *International Journal of Computer Vision*, vol. 73, no. 3, pp. 307–324, 2007.

[143] K. Gallivan, A. Srivastava, X. Liu, and P. Van Dooren, "Efficient algorithms for inferences on grassmann manifolds," in *IEEE Workshop on Statistical Signal Processing*, 2003.

[144] L. Latecki, R. Lakamper, and T. Eckhardt, "Shape descriptors for non-rigid shapes with a single closed contour," in *CVPR*, 2000.

[145] E. Begelfor and M. Werman, "Affine invariance revisited," in *CVPR*, 2006.

[146] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *CVPR*, 2009.

[147] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack, "Efficient color histogram indexing for quadratic form distance functions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 7, pp. 729–736, 1995.

[148] D. G. Lowe, "Object recognition from local scale-invariant features," *ICCV*, p. 1150, 1999.

[149] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 886–893, 2005.

[150] E. Chávez, G. Navarro, R. Baeza-Yates, and J. Marroquín, "Searching in metric spaces," *ACM Computing Surveys (CSUR)*, vol. 33, no. 3, pp. 273–321, 2001.

[151] G. Shakhnarovich, T. Darrell, and P. Indyk, *Nearest-neighbor methods in learning and vision*. MIT Press, 2006.

[152] G. Hjaltason and H. Samet, "Index-driven similarity search in metric spaces (survey article)," *ACM Transactions on Database Systems (TODS)*, vol. 28, no. 4, pp. 517–580, 2003.

[153] N. Kumar, L. Zhang, and S. Nayar, "What is a good nearest neighbors algorithm for finding similar patches in images?" in *European Conference on Computer Vision*, 2008, pp. 364–378.

[154] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proc. of 30th STOC*, 1998, pp. 604–613.

[155] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," *Proceedings of the International Conference on Very Large Data Bases*, pp. 518–529, 1999.

[156] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," *Proceedings of the Symposium on Computational geometry*, pp. 253–262, 2004.

[157] T. Darrell, P. Indyk, and G. E. Shakhnarovich, *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, 2006.

[158] M. Spivak, *A Comprehensive Introduction to Differential Geometry*, 3rd ed. Houston, Texas: Publish or Perish, Inc., 1999, vol. One.

[159] V. Athitsos, M. Potamias, P. Papapetrou, and G. Kollios, "Nearest neighbor retrieval using distance-based hashing," in *IEEE International Conference on Data Engineering*, 2008.

[160] J. Bourgain, "On lipschitz embedding of finite metric spaces in hilbert space," *Israel Journal of Mathematics*, vol. 52, no. 1, pp. 46–52, March 1985.

[161] X. Wang, J. T. Wang, K.-I. Lin, D. Shasha, B. A. Shapiro, and K. Zhang, "An index structure for data mining and clustering," *Knowledge and Information Systems*, vol. 2, no. 2, pp. 161–184, 2000.

[162] S. Dasgupta and Y. Freund, "Random projection trees and low dimensional manifolds," in *Proceedings of the 40th annual ACM symposium on Theory of computing*, 2008, pp. 537–546.

[163] R. Basri, T. Hassner, and L. Zelnik Manor, "Approximate nearest subspace search with applications to pattern recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[164] R. Chaudhry and Y. Ivanov, "Fast approximate nearest neighbor methods for non-euclidean manifolds with applications to human activity analysis in videos," in *ECCV*, 2010.

[165] A. Srivastava, I. Jermyn, and S. Joshi, "Riemannian analysis of probability density functions with applications in vision," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[166] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *Communications, IEEE Transactions on*, vol. 28, no. 1, pp. 84–95, 1980.

[167] N. Shroff, P. Turaga, and R. Chellappa, "Manifold Précis: An annealing technique for diverse sampling of manifolds," in *NIPS*, Dec. 2011.

[168] K. Fukunaga and M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *IEEE Transactions on Computing*, pp. 750–753, 1975.

[169] Y. Chikuse, *Statistics on special manifolds, Lecture Notes in Statistics*. Springer, New York., 2003.

[170] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *NIPS*, 2008.

[171] D. Marr, "Vision: A computational approach," 1982.

[172] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky, "Learning hierarchical models of scenes, objects, and parts," in *ICCV*, 2005.

[173] L. Li, R. Socher, and L. Fei-Fei, "Towards Total Scene Understanding: Classification, Annotation and Segmentation in an Automatic Framework," in *CVPR*, 2009.

[174] A. Gupta and L. Davis, "Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers," in *ECCV*, 2008.

[175] I. Biederman, "Aspects and extensions of theory of human image understanding," *Computational processes in human vision: An interdisciplinary perspective*, pp. 370–428, 1988.

[176] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, 2001.

[177] M. Gorkani and R. Picard, "Texture orientation for sorting photos "at a glance"," *TR-292, MIT, Media Labortory, Perceptual Computing Section*, 1994.

[178] S. Ullman, *The interpretation of visual motion*. MIT press Cambridge, MA, 1979.

[179] M. Marszałek, I. Laptev, and C. Schmid, "Actions in context," in *CVPR*, 2009.

[180] M. Szummer and R. Picard, "Indoor-outdoor image classification," in *1998 International Workshop on Content-Based Access of Image and Video Databases*, 1998, p. 42.

[181] A. Vailaya, M. Figueiredo, A. Jain, and H. Zhang, "Image classification for content-based indexing," *IEEE Transactions on Image Processing*, vol. 10, no. 1, pp. 117–130, 2001.

[182] L. Fei-Fei and P. Perona, "A Bayesian hierarchical model for learning natural scene categories," in *CVPR*, 2005.

[183] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*, 2006.

[184] A. Quattoni and A. Torralba, "Recognizing Indoor Scenes," in *CVPR*, 2009.

[185] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto, "Dynamic textures," *International Journal of Computer Vision*, 2003.

[186] M. Perc, "The dynamics of human gait," *European journal of physics*, vol. 26, no. 3, pp. 525–534, 2005.

[187] A. Chan and N. Vasconcelos, "Layered dynamic textures," *Advances in Neural Information Processing Systems*, 2006.

[188] R. Vidal and A. Ravichandran, "Optical flow estimation and segmentation of multiple moving dynamic textures," in *CVPR*, 2005.

[189] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering object category in image collection," in *ICCV*, 2005.

[190] S. Ali, A. Basharat, and M. Shah, "Chaotic Invariants for Human Action Recognition," in *ICCV*, 2007.

[191] G. Williams, *Chaos theory tamed*. Taylor & Francis Ltd, 1997.

[192] A. Fraser and H. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, vol. 33, no. 2, pp. 1134–1140, 1986.

[193] M. Rosenstein, J. Collins, C. De Luca *et al.*, "A practical method for calculating largest Lyapunov exponents from small data sets," *Physica D*, vol. 65, no. 1-2, pp. 117–134, 1993.

[194] K. De Cock and B. De Moor, "Subspace angles and distances between ARMA models," in *Proc. of the Intl. Symp. of Mathematical Theory of Networks and Systems*, vol. 1, 2000.

[195] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar, "Attribute and Simile Classifiers for Face Verification," in *ICCV*, 2009.

[196] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *CVPR*, 2009.

[197] C. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *CVPR*, 2009.

[198] L. F. Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *CVPR Workshop on Generative Model Based Vision*, 2004.

[199] N. Shroff, A. Veeraraghavan, Y. Taguchi, O. Tuzel, A. Agrawal, and R. Chellappa, "Variable focus video: Reconstructing depth and video for dynamic scenes," in *International Conference on Computational Photography*, 2012.

[200] Z. Myles and N. D. V. Lobo, "Recovering affine motion and defocus blur simultaneously," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1996.

[201] S. M. Seitz and S. Baker, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[202] N. Shroff, Y. Taguchi, O. Tuzel, A. Veeraraghavan, S. Ramalingam, and H. Okuda, "Finding a needle in a specular haystack," in *ICRA*, 2011.

[203] N. Shroff, P. Turaga, and R. Chellappa, "Moving vistas: Exploiting motion for describing scenes," in *CVPR*, 2010.

[204] J. Yuan, Z. Liu, and Y. Wu, "Discriminative subvolume search for efficient action detection," in *CVPR*, 2009.

[205] W. Yang, Y. Wang, and G. Mori, "Efficient human action detection using a transferable distance function," in *ACCV*, 2010.

[206] V. Valdés and J. Martínez, "On-line video summarization based on signature-based junk and redundancy filtering," in *Ninth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2008.

[207] ——, "Binary tree based on-line video summarization," in *Proceedings of the 2nd ACM TRECVid Video Summarization Workshop*, 2008.

[208] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 515–528, 2003.

[209] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *SIAM International Conference on Data Mining*, 2006.

[210] G. Golub and C. Van Loan, *Matrix computations*. Johns Hopkins University Press, 1996.